Уральский федеральный университет
имени первого Президента России Б.Н.Ельцина
Институт фундаментального образования

О. В. ЛИМАНОВСКАЯ
Т. И. АЛФЕРЬЕВА

МОДЕЛИРОВАНИЕ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ В ANYLOGIC 8.1

Лабораторный практикум

Министерство науки и высшего образования
Российской Федерации
Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина

О. В. Лимановская, Т. И. Алферьева

Моделирование производственных процессов в AnyLogic 8.1

Лабораторный практикум

Рекомендовано методическим советом
Уральского федерального университета
для студентов вуза, обучающихся
по направлению подготовки
09.03.04 — Программная инженерия

Екатеринбург
Издательство Уральского университета
2019

УДК 004.94(076.5)
ББК 32.972в6я73-5
Л58

Рецензенты:

д-р пед. наук, проф. *Л. И. Долинер* (завкафедрой информационных систем и технологий Уральского технического института связи и информатики (филиал СибГУТИ));

канд. физ.-мат. наук, науч. сотр. Института высокотемпературной электрохимии *А. Н. Езин*

Научный редактор доц., канд. хим. наук *Н. А. Хлебников*

Лимановская, О. В.

Л58 Моделирование производственных процессов в AnyLogic 8.1 : лабораторный практикум / О. В. Лимановская, Т. И. Алферьева. — Екатеринбург : Изд-во Урал. ун-та, 2019. — 136 с.

ISBN 978-5-7996-2680-8

Издание представляет собой лабораторный практикум, позволяющий освоить работу в среде AnyLogic с потоками и со сложными моделями, включающий в себя несколько подходов моделирования. Предназначено для студентов-бакалавров, магистров и аспирантов всех форм обучения, обучающихся по техническим специальностям.

Рис. 202.

УДК 004.94(076.5)
ББК 32.972в6я73-5

ISBN 978-5-7996-2680-8

Введение

В настоящее время практически любой новый проект начинается с моделирования его процессов и поиска оптимального решения во время эксперимента с моделью. Моделирование широко применяется как в бизнесе, так и в научных исследованиях. Но если в научных исследованиях, как правило, используется аналитическое моделирование, основу которого составляют системы уравнений различного вида, то при моделировании бизнес-процессов или производств часто невозможно описать систему набором систем уравнений. Тогда на помощь приходит имитационное моделирование.

Имитационное моделирование основывается на наборе состояний системы, переходы между которыми задаются событиями системы, т. е. переход из текущего состояния в последующее заранее не известен и зависит от того события, которое может произойти. Для примера рассмотрим модель работы отделения банка. Допустим, изначально система находится в состоянии ожидания. Далее, если пришел клиент и выбрал услугу получения кредита, то система перейдет в состояние оценки кредитоспособности клиента. Если клиент выберет услугу оплаты государственной пошлины, то система перейдет в состояние оказания услуги оплаты государственной пошлины. Если клиент вообще не придет, то система остается в состоянии ожидания.

Традиционно имитационное моделирование представляется тремя подходами:

1) дискретно-событийным;
2) агентным;
3) системной динамикой.

В *дискретно-событийном подходе* выделяется пассивный объект моделирования — заявка и активный субъект — сервис. В качестве заявки могут быть рассмотрены клиенты банка, посетители кафе, детали на производстве и т. д. Сервисом служат операции, которые выпол-

няются над заявками — обслуживание клиентов, посетителей, обработка деталей и т. д. Сервисы используют ресурсы для своей работы. Под ресурсами понимаются как служащий персонал, так и оборудование, необходимое для процесса, в том числе и помещение, если его необходимо учитывать в модели. Такой подход применим для моделирования процессов с большой детализацией, когда важно поведение каждой заявки. Например, этот подход хорошо работает для моделирования логистики склада.

Системная динамика основана на построении причинно-следственных связей и напрямую не является разделом имитационного моделирования. В настоящее время системная динамика получила широкое распространение для моделирования маркетинговых компаний, политики банка, страховых стратегий. В ней выделяется накопитель (уровень) и поток. Уровень отображает как причину, так и следствие, но переход причины в следствие задается потоком. Поток представляет собой фактически производную по времени уровня причины. Например, причиной может быть уровень удовлетворенности клиентов банка, а следствием — количество обращений в банк.

Агентный подход является универсальным инструментом и основан на понятии агента. Агент — это элемент системы, который имеет свои параметры, методы и поведение. Параметры агента задаются переменными и представляют собой некие характеристики агента. Методы агента представляют собой действия агента и задаются функциями. Поведение агента представляет собой набор состояний агента, связанных переходами между собой, реализуется как конечный автомат.

При моделировании можно использовать любой из перечисленных подходов, но условия задачи, как правило, делают удобным какой-либо конкретный из них. В то же время реальные системы содержат множество задач и не могут укладываться в применение одного подхода, поэтому большинство моделей реальных систем (производств, логистических цепочек и т. д.) представляют собой сложные модели, в которых использованы несколько подходов.

Настоящее пособие посвящено изучению построения многоподходных моделей в среде имитационного моделирования AnyLogic.

Лабораторная работа № 1

Разработка модели технологической сборки изделия

Задача

Промоделировать работу технологической цепочки по сборке изделия, состоящего из двух деталей. Первая деталь изделия подвергается двум технологическим операциям до сборки, вторая деталь изделия подвергается одной технологической операции до сборки. Первая технологическая операция над первой деталью длится от 3 до 5 минут и выполняется 1 роботом. Вторая технологическая операция с первой деталью длится от 4 до 8 минут и выполняется 1 рабочим, который работает согласно расписанию (с 8 до 17 по рабочим дням с перерывом на обед с 12 до 13). Технологическая операция по обработке второй детали длится от 6 до 10 минут и выполняется рабочим. Сборка изделия выполняется роботом и длится от 6 до 12 минут. Изделие после сборки упаковывается по 5 штук. Упаковка изделий осуществляется рабочим и длится от 10 до 16 минут. Первая деталь для сборки поставляется со склада1 в количестве 1 штуки в час. Вторая деталь для сборки поставляется со склада2 в количестве 2 штуки в час.

Решение

Для решения поставленной задачи будет использоваться дискретно-событийный подход. В этом подходе рассматривается заявка-агент, которую обслуживают на различных операциях. В качестве заявки-агента в данной задаче рассматриваются детали и само изделие. Технологические операции, выполняемые над деталями, рассматриваются как обслуживание заявки-агента различными сервисами.

После запуска программы AnyLogic откроется окно с начальной страницей, на которой содержится справочная информация по программе (рис. 1.1).

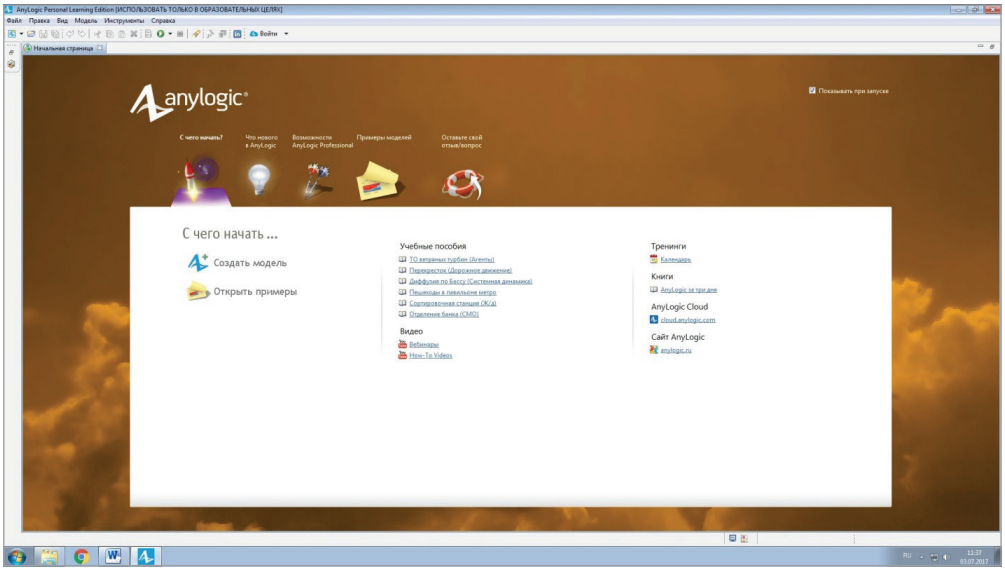

Рис. 1.1. Начальная страница

Это окно в дальнейшей работе не нужно, поэтому его можно закрыть. В рабочем окне программы выберите из строки меню **Файл** → **Создать** → **Модель** (рис. 1.2).

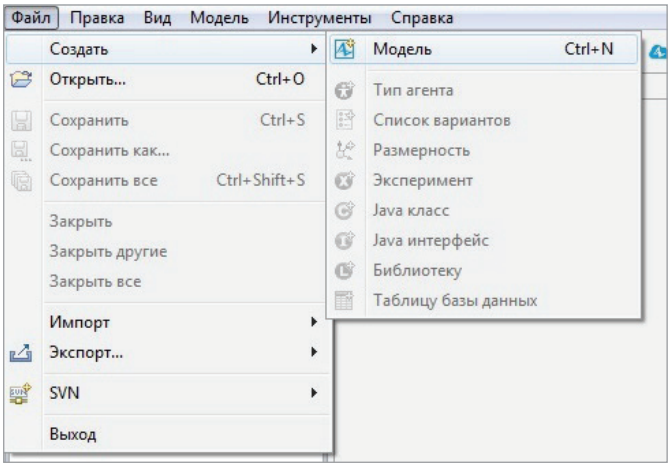

Рис. 1.2. Создание модели

В открывшемся окне мастера создания модели задайте имя модели и единицы модельного времени — **минуты** (рис. 1.3).

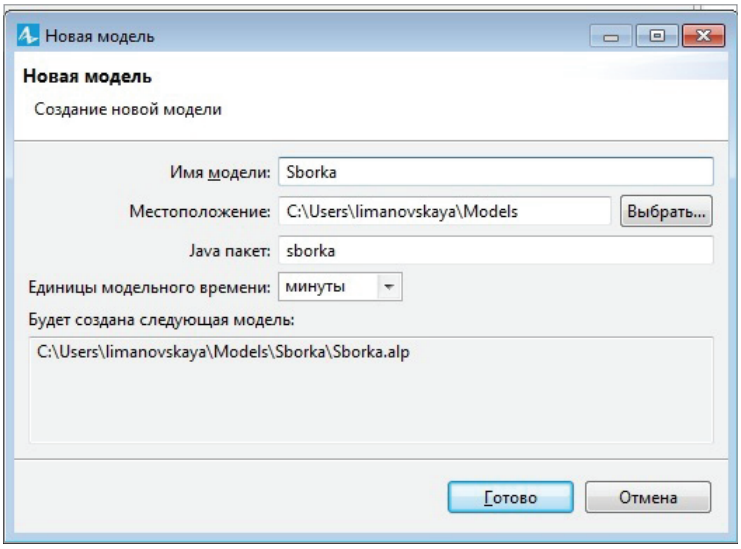

Рис. 1.3. Задание параметров модели

Откроется рабочее окно модели (рис. 1.4), которое разделено на три части. В левой части находятся закладки **Проекты** и **Палитра**.

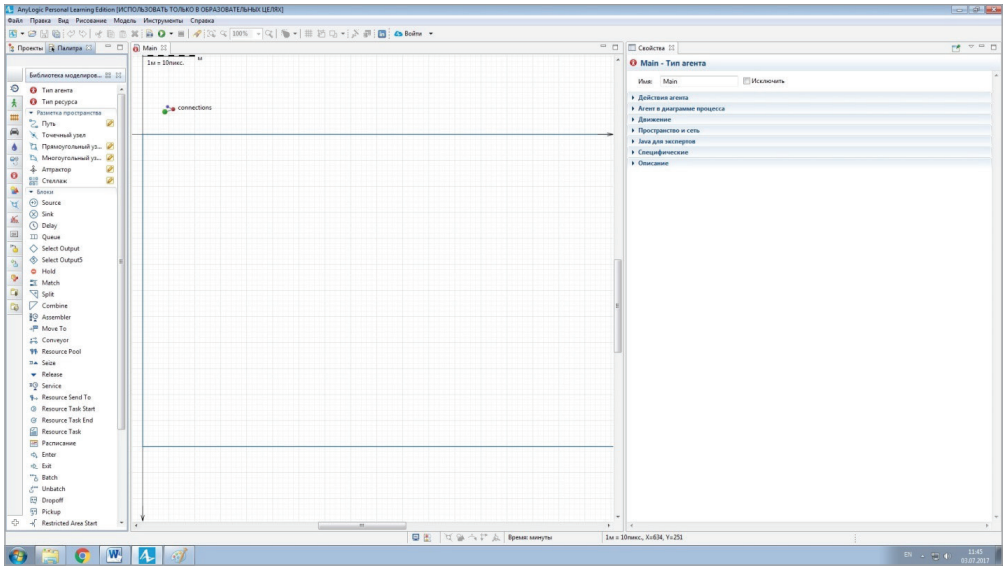

Рис. 1.4. Рабочее окно модели

В закладке **Проекты** представлены все открытые проекты и их содержимое. В закладке **Палитра** находятся все инструменты моделирования, которые разделены на разные библиотеки. Все необходимые инструменты для дискретно-событийного моделирования находятся в **Библиотеке моделирования процессов**, которая будет автоматически открываться. Средняя зона представляет собой рабочее поле, в котором будет собираться модель. В правой части отображаются свойства **выделенного в данный момент** элемента модели.

Этап 1. Моделирование агентов — деталей и изделия

Детали, из которых будет производиться сборка и само изделие, будут представлены в модели как **Тип агентов**. Перетащите ярлык **Тип агента** из библиотеки на рабочее поле. Откроется окно мастера создания агента (рис. 1.5).

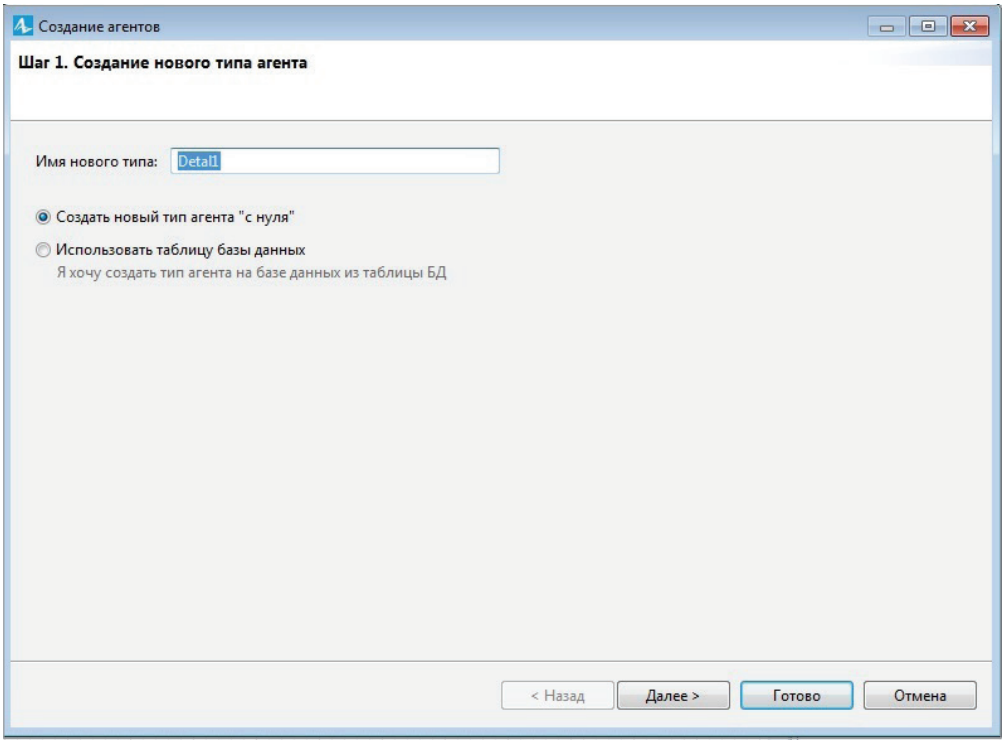

Рис. 1.5. Окно мастера создания агента

Задайте в нем имя агента и нажмите **Готово**. После этого автоматически откроется окно агента. Закройте его. Повторите операцию для каждой детали и изделия. Имена агентов — для детали1 — Detal1, для детали2 — Detal2, для изделия — Isdelie. После всех операций на вкладке **Проекты** должен быть список из 4 агентов: **Detal1**, **Detal2**, **Isdelie**, **Main** (рис. 1.6).

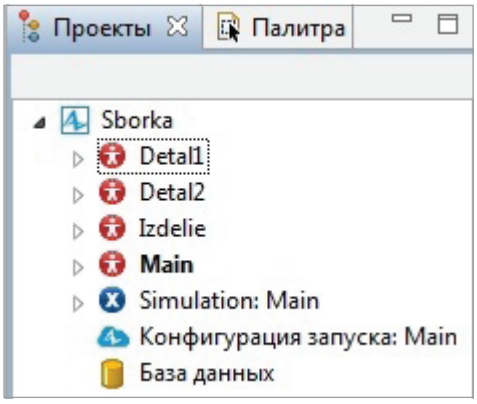

Рис. 1.6. Список агентов модели

Этап 2. Моделирование поставок деталей

Для того чтобы моделировать появление заявок-агентов в модели, необходимо использовать блок **Source** (рис. 1.7)

Рис. 1.7. Блок **Source**

Для моделирования поставок деталей1 перетащите блок **Source** на рабочее поле модели. Задайте в свойствах блока имя склада, интенсивность поставки 1 в час и в разделе **Новый агент** — тип агента **Detal1** (рис. 1.8).

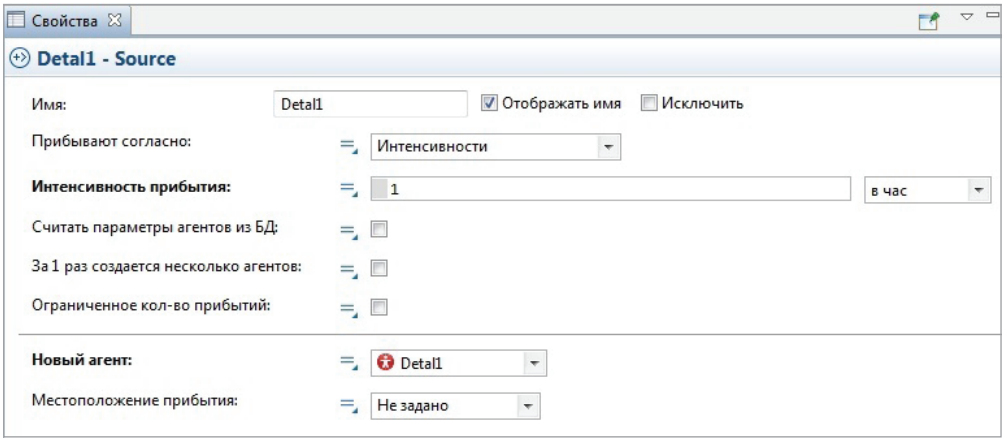

Рис. 1.8. Задание свойств склада деталей1

Повторите те же операции для деталей2, задав интенсивность прибытия 2 в час и выбрав в разделе **Новый агент** тип агента **Detal2** (рис. 1.9).

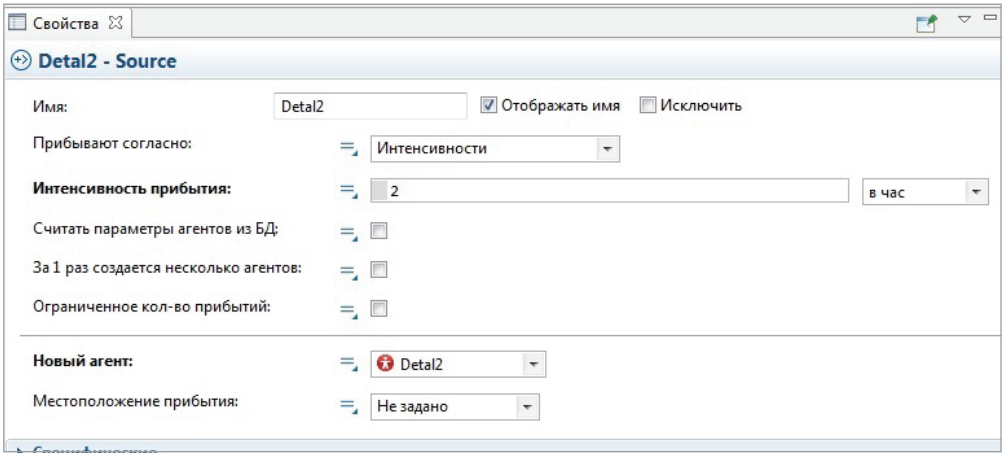

Рис. 1.9. Задание свойств склада деталей2

Этап 3. Моделирование обработки деталей1

Первая деталь подвергается двум технологическим операциям. Каждая операция рассматривается как обслуживание агента **Detal1**. Для обслуживания агента используется блок **Service** (рис. 1.10).

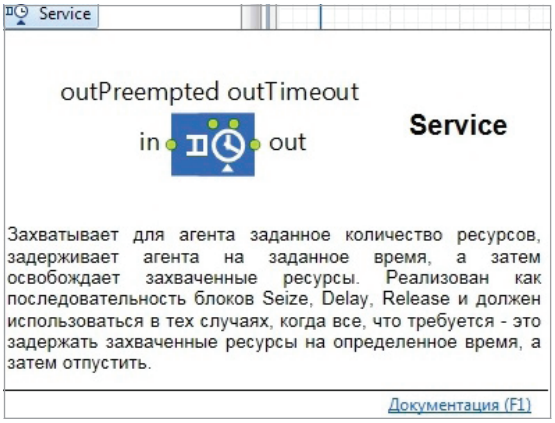

Рис. 1.10. Блок **Service**

Данный блок состоит из очереди на обслуживание и блока, имитирующего задержку детали на время обслуживания. Блок имеет один вход и три выхода: **out**, **outTimeout**, **outPreempted**. Из выхода **out** выходят те детали, чье обслуживание было завершено. Из выхода **outTimeout** выходят те детали, чье время ожидания обслуживания в очереди истекло. Из выхода **outPreempted** выходят детали, вытесненные деталями с более высоким приоритетом на обработку.

В данной задаче все детали должны пройти обслуживание, поэтому будет использоваться только выход **out**.

Задание первой технологической операции

Перетащите блок **Service** на рабочее поле модели так, чтобы его вход соединился с выходом блока **Detal1** (рис. 1.11).

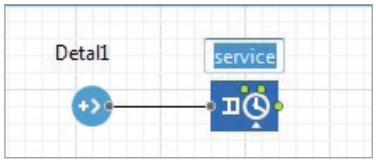

Рис. 1.11. Перенос блока **Service** на рабочее поле модели

В свойствах блока задайте его имя (**Operaciya1**), вместимость очереди — **Максимальная вместимость**, длительность операции (рис. 1.12). Длительность операции задается треугольным распределением, в котором первый параметр означает минимальное время операции (3 минуты), второй параметр — среднюю продолжительность операции (4 минуты), третий — максимальную продолжительность операции (5 минут).

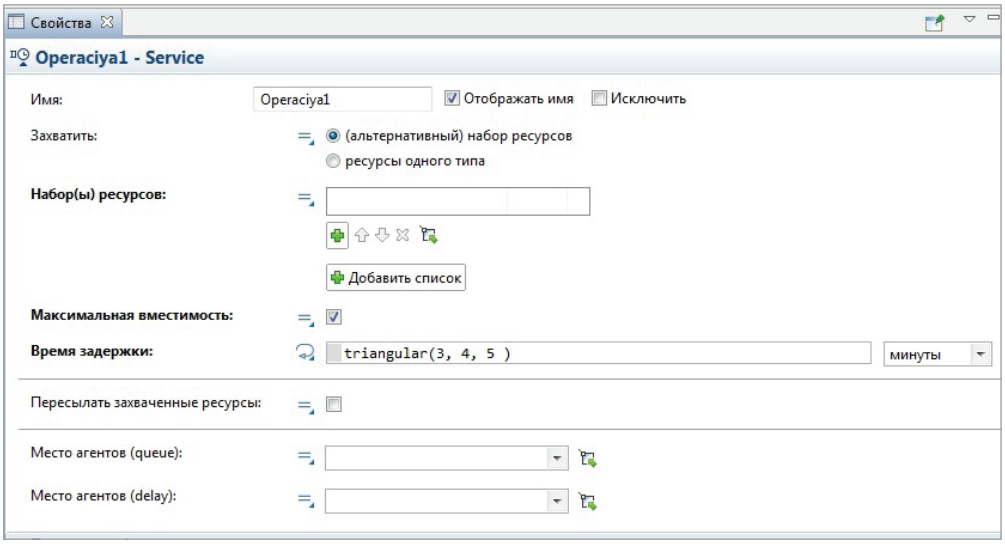

Рис. 1.12. Задание свойств операции1

Задание ресурсов для выполнения 1-й технологической операции

Как было сказано в задании, первую технологическую операцию с деталью1 выполняет робот. Для моделирования задания операции для робота нужно в блоке **Service** указать **Набор ресурсов**, который выполняет операцию. Чтобы указать **Набор ресурсов**, нужно заранее создать ресурсы в модели.

Ресурсы в модели задаются блоком **Resource Pool** (рис. 1.13).

Перетащите блок **Resource Pool** на рабочее поле модели. В его свойствах (рис. 1.14) задайте имя ресурса **Robot1**, количество и тип — **Переносной**. Вообще, ресурсы в модели могут быть трех типов: движущийся, переносной и статический. Если выбрать движущийся тип, то робот может обслуживать несколько операций, передвигаясь от одной к другой.

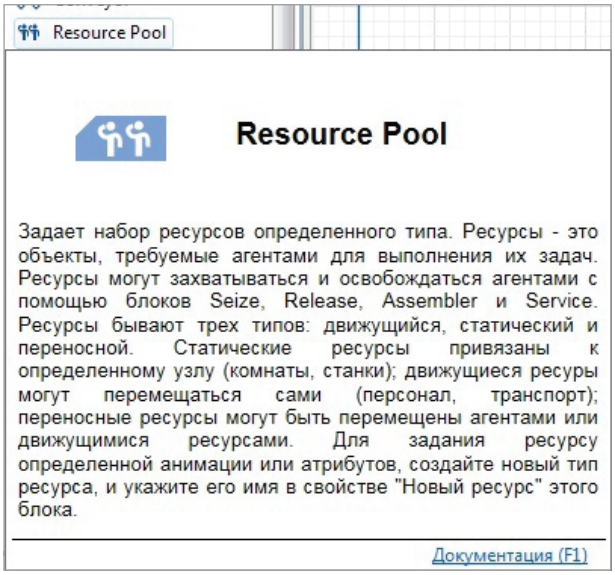

Рис. 1.13. Блок **Resource Pool**

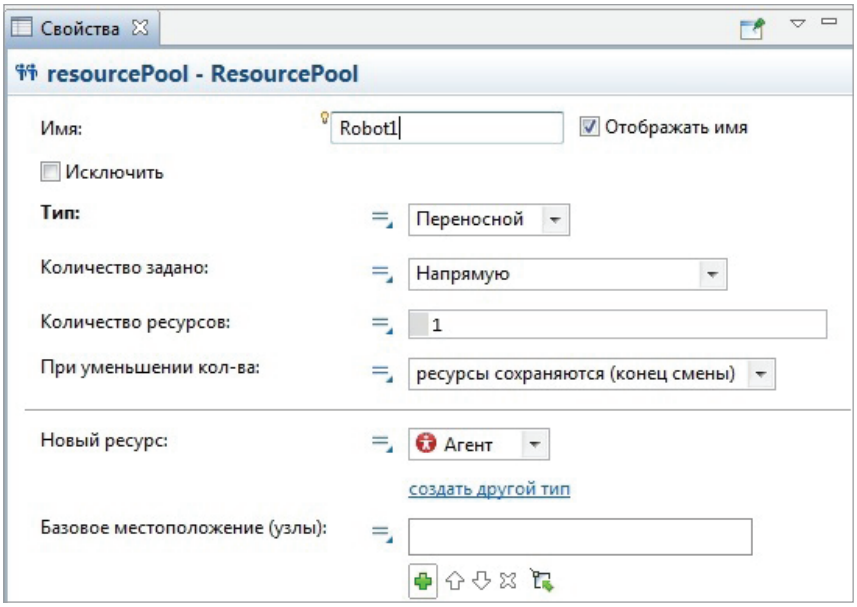

Рис. 1.14. Свойства ресурса робот1

Теперь можно добавить ресурс **Robot1** к первой технологической операции. Для этого перейдите в свойства блока **Operaciya1** и в разделе **Набор ресурсов** нажмите на + и добавьте ресурс **Robot1** (рис. 1.15).

Рис. 1.15. Добавление ресурса **Robot1**

В результате в свойствах блока **Operaciya1** должно получиться, как на рис. 1.16.

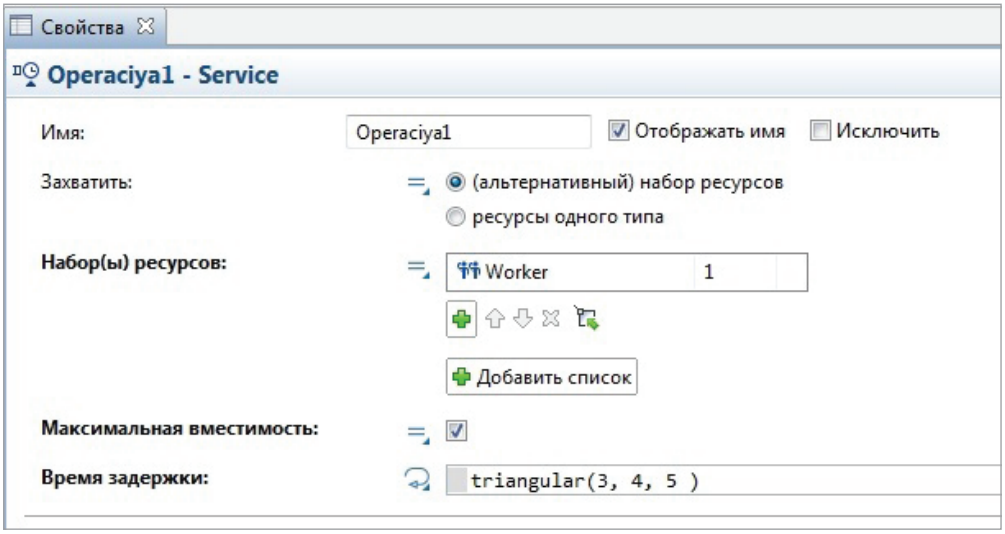

Рис. 1.16. Свойства **Operaciya1**

Задание второй технологической операции

Перетащите блок **Service** так, чтобы выход **out** блока **Operaciya1** соединился со входом нового блока (рис. 1.17).

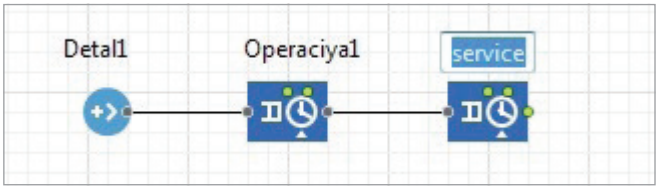

Рис. 1.17. Задание второй операции

В свойствах блока задайте его имя **Operaciya2**, вместимость очереди 10 и время выполнения — 4, 6 и 8 минут (рис. 1.18).

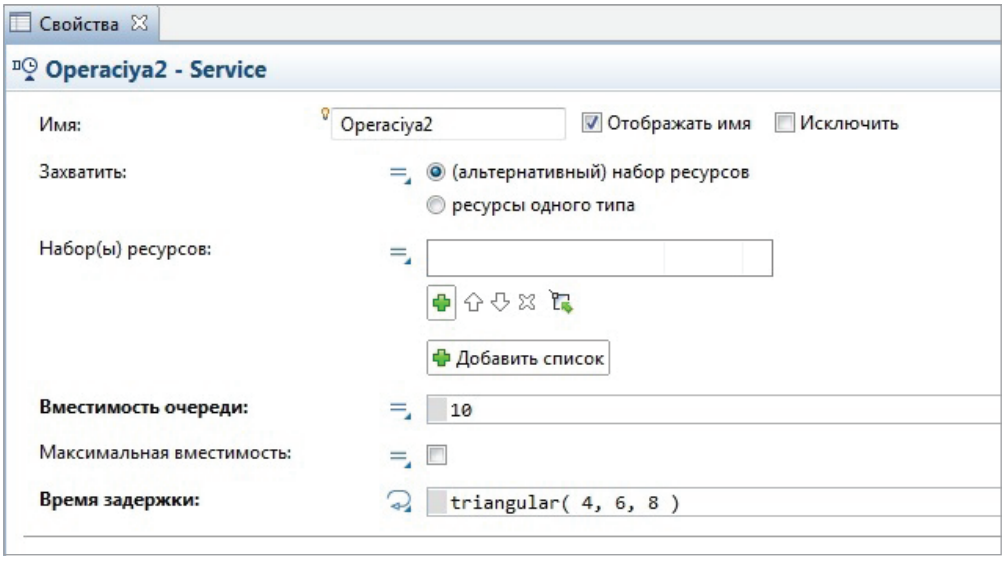

Рис. 1.18. Задание свойств второй операции

Задание ресурсов для выполнения второй технологической операции

Вторую технологическую операцию выполняет рабочий, который работает по заданному расписанию. Для задания расписания работы рабочего используется блок **Расписание** (рис. 1.19).

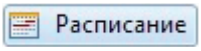

Рис. 1.19. Блок **Расписание**

Перетащите блок **Расписание** на рабочее поле модели. В свойствах блока **Расписание** задайте два расписания — с 8 до 12 и с 13 до 17. Добавить расписание необходимо с помощью кнопки **+** справа от таблицы **Расписание**. Тип расписания задайте **да/нет** (показывает информацию о том, занят или свободен ресурс, работающий по данному расписанию, рис. 1.20).

Далее, задайте ресурс с помощью блока **Resource Pool**. Поскольку работа рабочего задается расписанием, то в свойствах блока **Resource Pool** в пункте **Количество задано** выбираем пункт **Расписание доступности**. Если выбрать просто пункт **Расписание**, то нужно будет в самом блоке **Расписание** задавать количество рабочих, занятых в указанные часы. Поскольку в блоке **Расписание** выбран тип расписания **да/нет**, т. е. доступен или нет ресурс, то этот тип расписания называется **Рас-**

писанием доступности. В появившемся пункте **Расписание доступности** выберите только что созданное расписание. В итоге свойства блока **ResourcePool** должны выглядеть так, как на рис. 1.21.

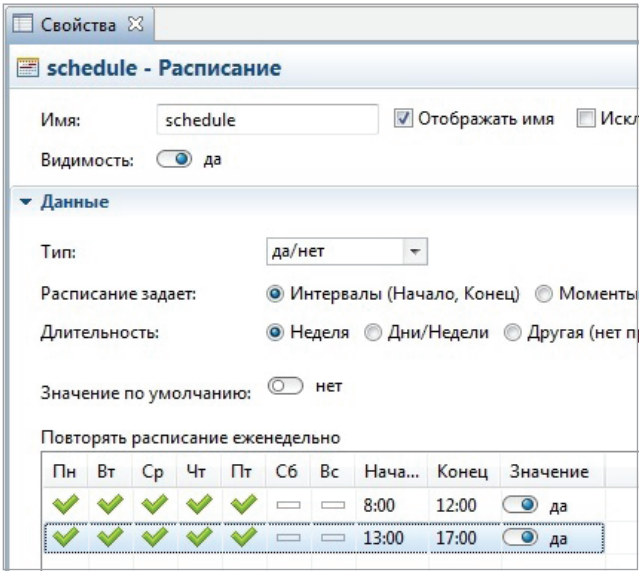

Рис. 1.20. Задание расписания работы рабочего

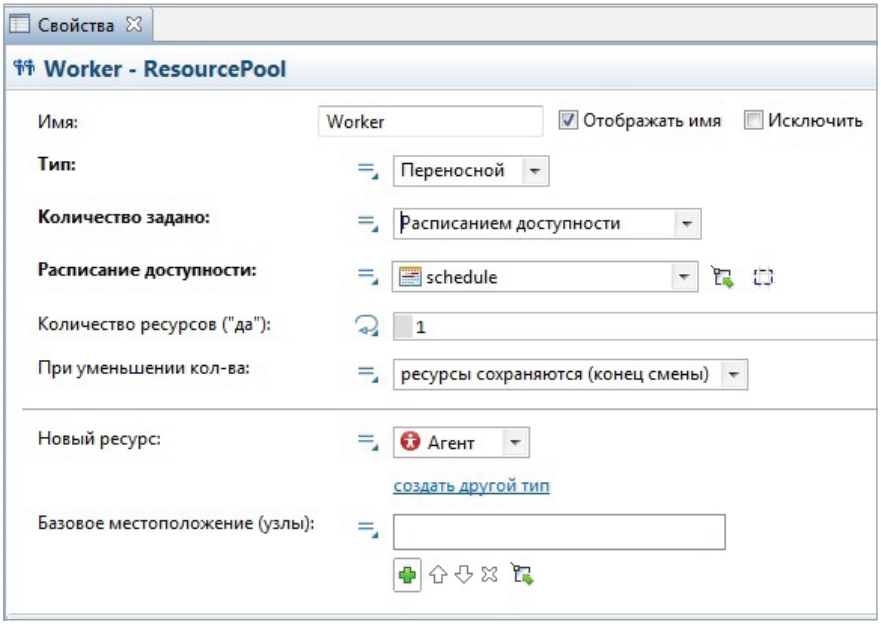

Рис. 1.21. Свойства ресурса **Рабочий**

Добавьте ресурс ко второй операции (рис. 1.22).

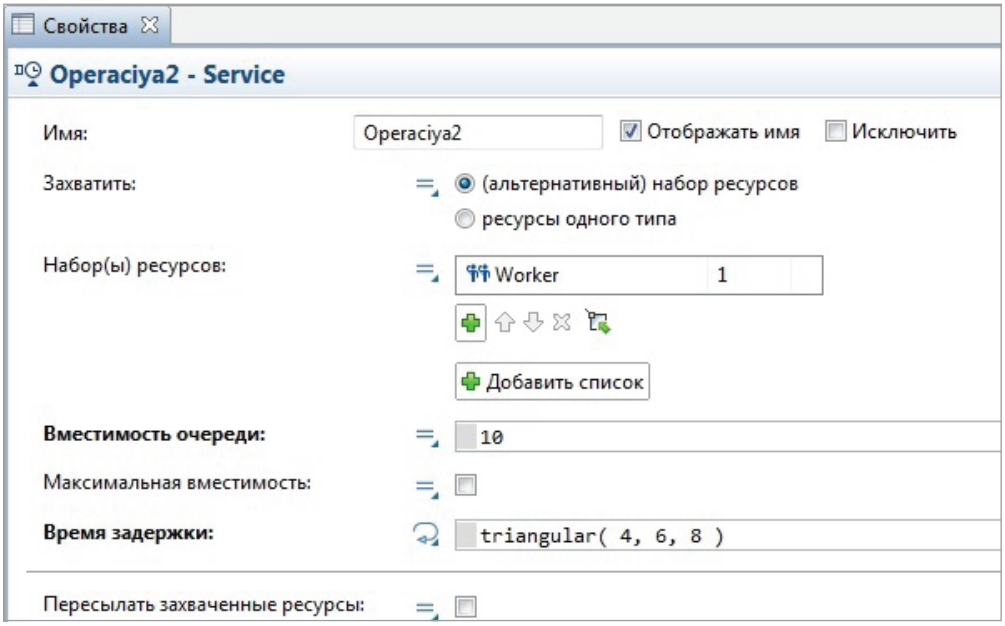

Рис. 1.22. Свойства второй операции с добавленным ресурсом

Задание операции обработки второй детали

Промоделируем выполнение операции с помощью блока **Service**. Перетащите блок **Service** на рабочее поле модели так, чтобы его вход соединился с выходом блока **Detal2** (рис. 1.23).

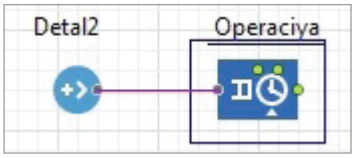

Рис. 1.23. Присоединение блока **Operaciya**

В свойствах блока задайте его имя (**Operaciya**), вместимость очереди (**Максимальная**) и время выполнения операции (рис. 1.24).

Технологическая операция выполняется рабочим. Причем это другой рабочий, не тот, который обрабатывает первую деталь. Поэтому нужно создать новый ресурс. А вот работать этот рабочий будет по тому же расписанию, что и рабочий, обрабатывающий первую деталь, поэтому создавать новое расписание не нужно. Итак, создайте

новый ресурс, работающий по имеющемуся расписанию. Для этого перетащите блок **Resource Pool** на рабочее поле модели и задайте его свойства (рис. 1.25).

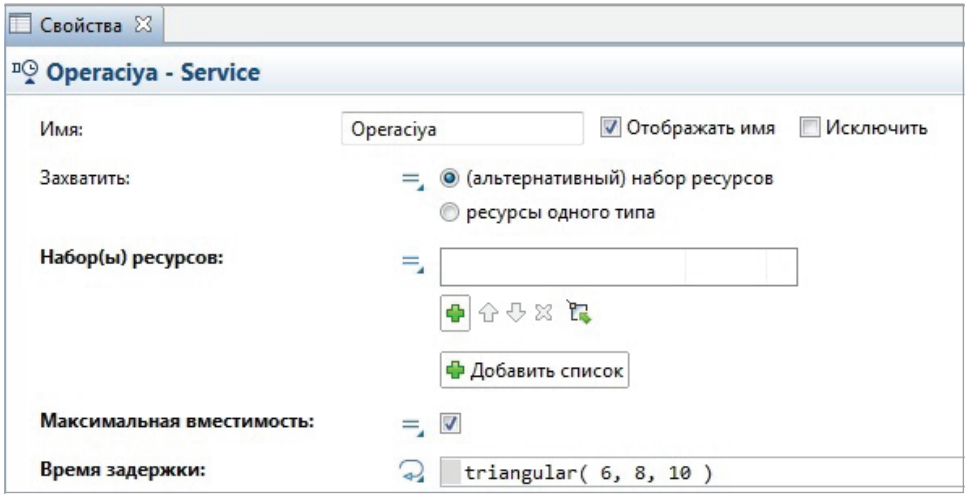

Рис. 1.24. Свойства блока **Operaciya**

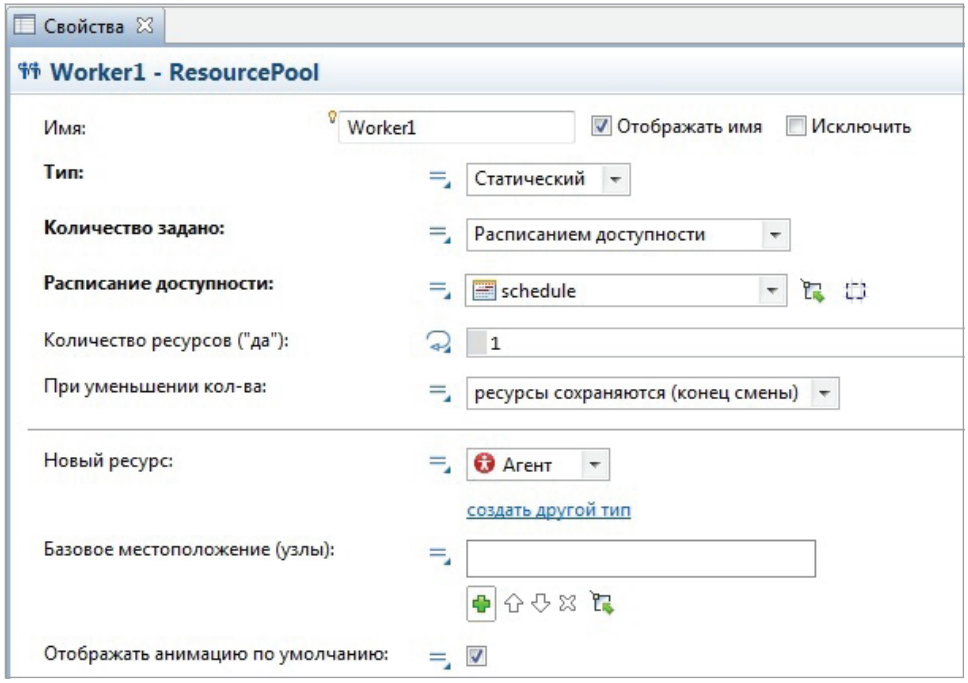

Рис. 1.25. Свойства блока **Worker1**

Теперь добавьте ресурсы к операции обработки детали, зайдя в свойства блока **Opraciya** и добавив **Набор ресурсов** (рис. 1.26).

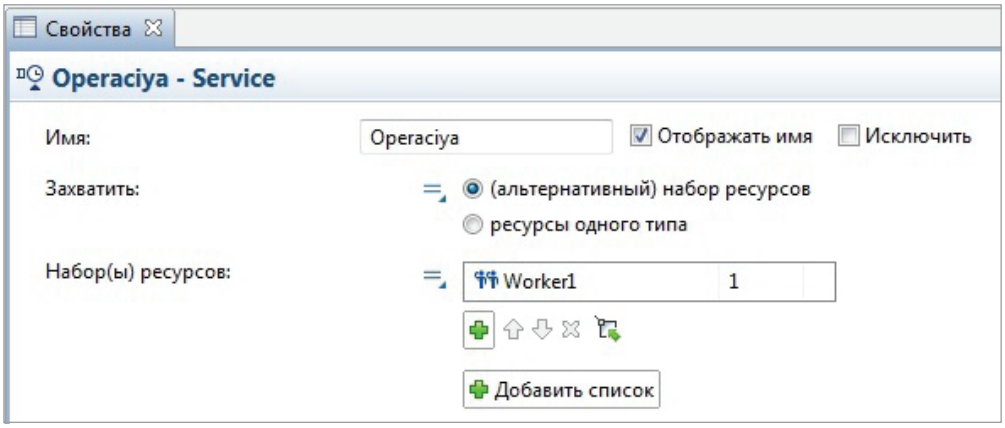

Рис. 1.26. Свойства блока **Opraciya** с добавленными ресурсами

Задание операции сборки

Для моделирования процесса сборки в среде **AnyLogic** используется блок **Assembler**, который имеет 5 входов и один выход (рис. 1.27). На вход в него подаются детали, из которых собирается изделие, на выходе из блока получается новый агент-заявка, а именно собранное изделие.

Рис. 1.27. Блок **Assembler**

Перетащите блок **Assembler** на рабочее поле модели и соедините его входы с выходами операций обработки деталей.

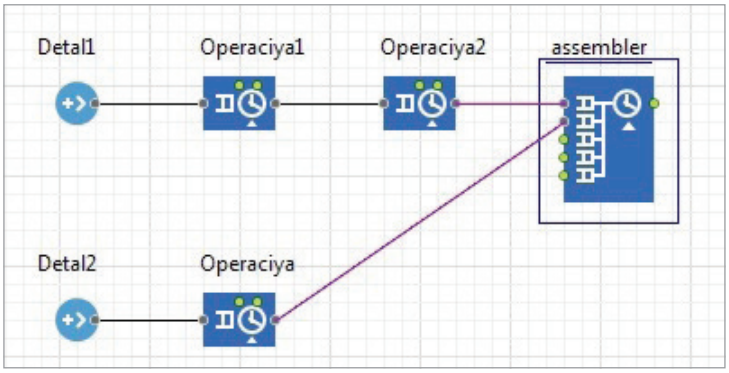

Рис. 1.28. Присоединение операции сборки

В свойствах блока **Assembler** задайте время выполнения сборки и тип агента на выходе в разделе **Новый агент** (рис. 1.29).

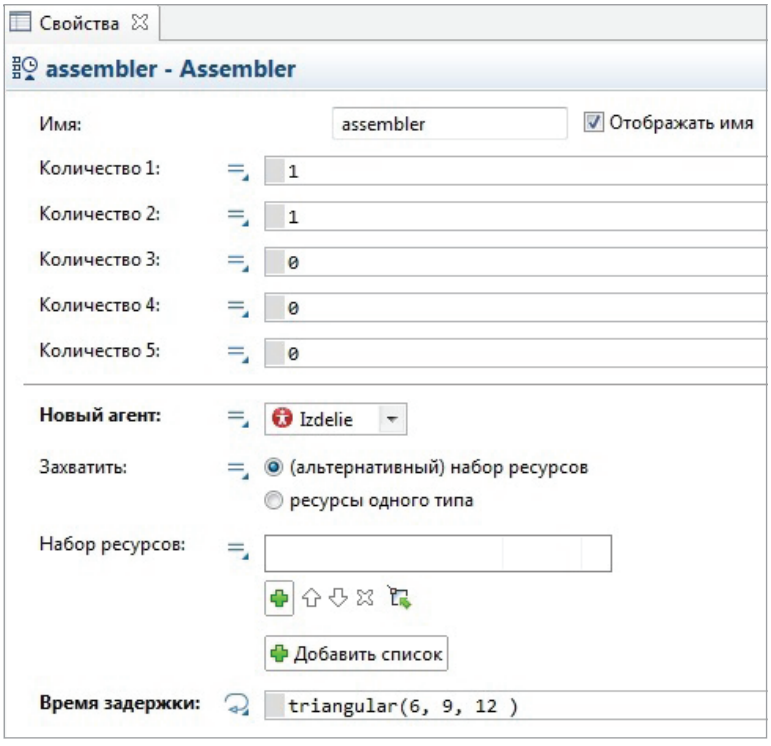

Рис. 1.29. Свойства блока **Assembler**

Сборку осуществляет робот, предназначенный именно для этой операции. Поэтому нужно с помощью блока **Resource Pool** создать еще один тип ресурсов **RobotAssembler**. Для этого перетащите блок **Resource Pool** на рабочее поле модели и задайте его свойства (рис. 1.30).

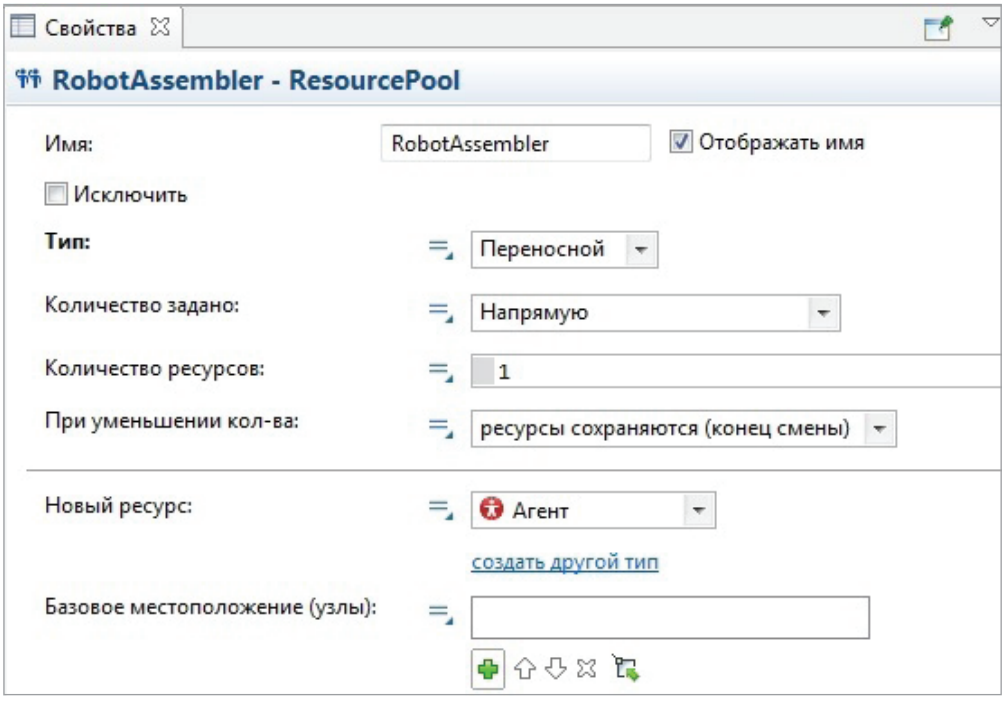

Рис. 1.30. Свойства блока **RobotAssembler**

Присоедините ресурс робот к процессу сборки (рис. 1.31).

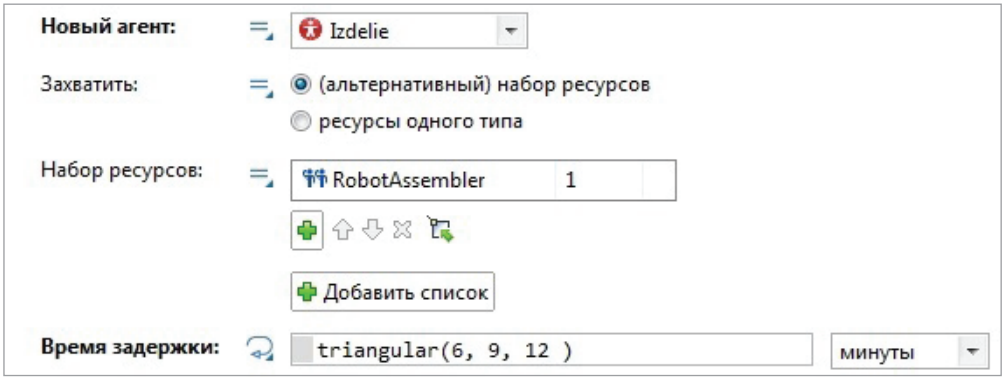

Рис. 1.31. Назначение ресурса для процесса сборки

Задание операции упаковки

Операция сборки также задается блоком **Service**, а рабочий, выполняющий ее, задается блоком **Resource Pool**. Поскольку этот рабочий работает по тому же расписанию, что и все остальные рабочие, то используется ранее созданное расписание доступности.

Перетащите блок **Resource Pool** на рабочее поле модели и задайте его свойства (рис. 1.32).

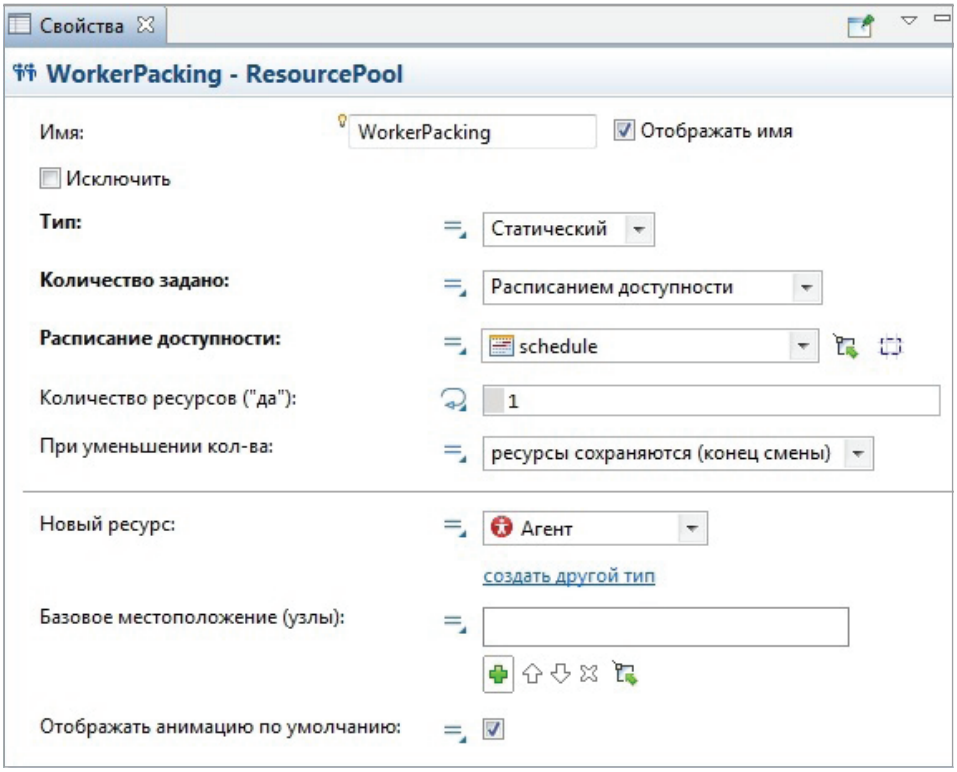

Рис. 1.32. Свойства блока **WorkerPacking**

Перетащите блок **Service** на рабочее поле модели так, чтобы его вход соединился с выходом блока **Assembler**, и задайте его свойства (рис. 1.33).

Из операции упаковки выходят коробки, содержащие по 5 изделий. Для моделирования коробок введем нового агента в модель — **Box**.

Перетащите на рабочее поле модели блок **Тип агента** и в открывшемся мастере создания агента введите его имя **Box**. Нажмите кнопку **Готово**. Закройте автоматически открывшееся окно агента **Box**.

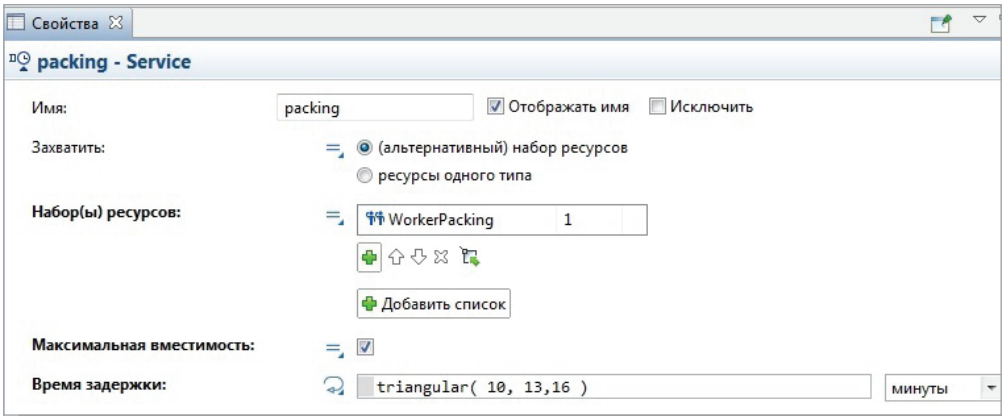

Рис. 1.33. Свойства блока **packing**

Упаковку изделий промоделируем с помощью блока **Batch** (рис. 1.34). Этот блок принимает на вход заданное количество изделий и выпускает их партию. Если задать постоянную партию, то ее уже нельзя будет разобрать на отдельные изделия.

Рис. 1.34. Блок **Batch**

Перетащите блок **Batch** на рабочее поле модели так, чтобы его вход соединился с выходом блока **packing** (рис. 1.35).

Задайте свойства блока (рис. 1.36).

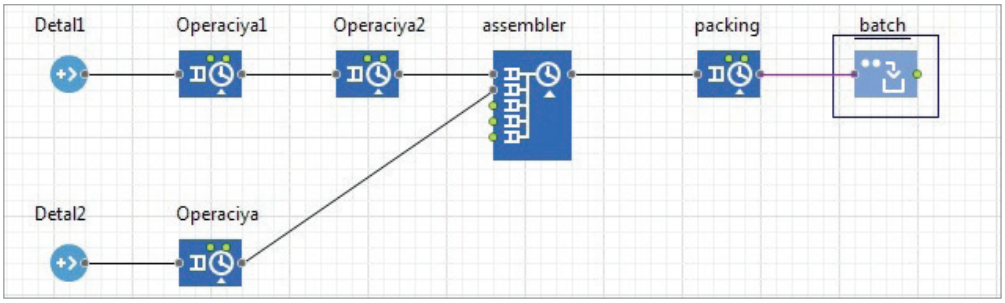

Рис. 1.35. Присоединение блока **Batch**

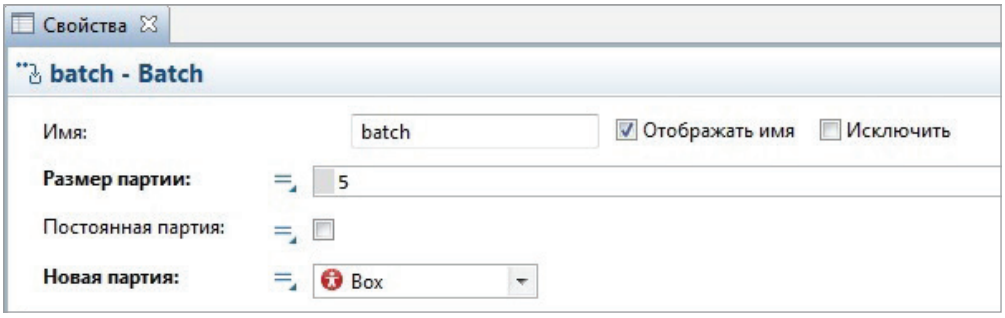

Рис. 1.36. Свойства блока **batch**

Собранные коробки с изделиями увозятся из цеха. Для моделирования ухода коробок из модели используется блок **Sink** (рис. 1.37). Он имеет один вход и просто уничтожает входящие в него заявки.

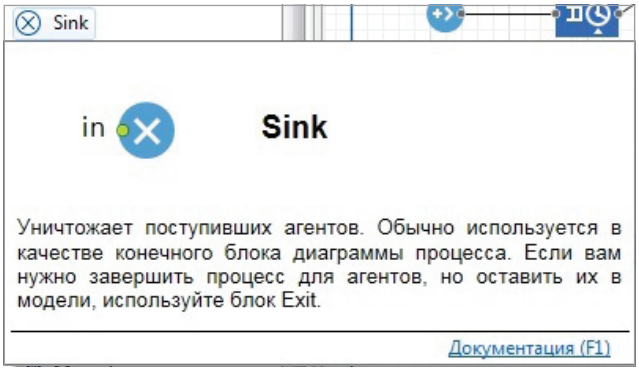

Рис. 1.37. Блок **Sink**

Перетащите блок **Sink** на рабочее поле модели так, чтобы его вход соединился с выходом блока **batch** (рис. 1.38).

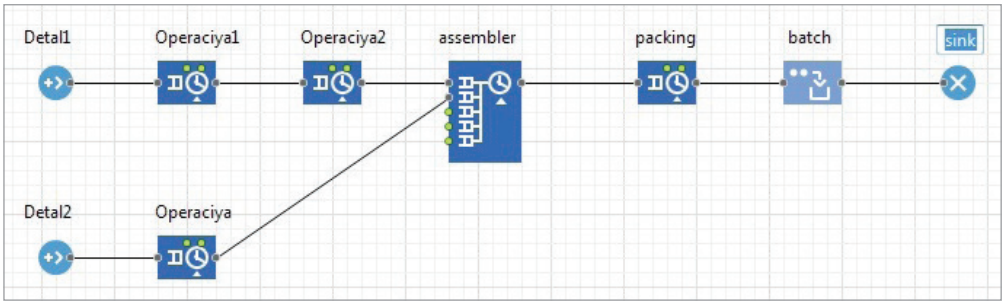

Рис. 1.38. Модель производства

Запустите полученную модель, нажав на кнопку  на панели инструментов в главном меню. Откроется окно запуска модели (рис. 1.39).

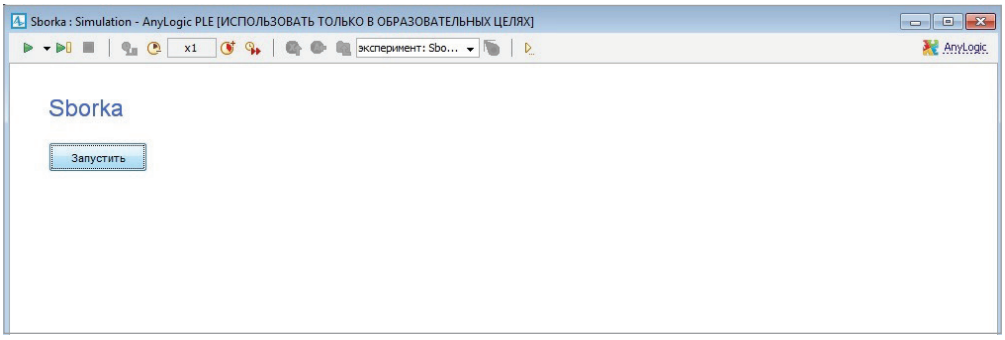

Рис. 1.39. Окно запуска модели

Нажмите на кнопку **Запустить** — откроется окно работающей модели (рис. 1.40).

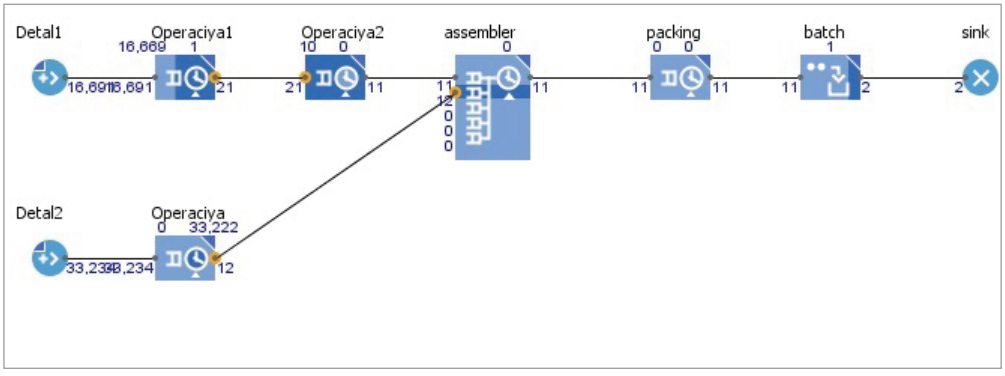

Рис. 1.40. Работа модели

Проблемные места в модели выделяются красным. Как видно из рис. 1.40, проблемы возникают на выходе из первой технологической операции и на входе во вторую технологическую операцию. Также есть проблемы на выходе из операции обработки второй детали. Поварьируете настройки модели, включая время выполнения сборки, чтобы получить наибольшую занятость робота-сборщика и ликвидировать затор из вторых деталей на входе в операцию сборки.

Лабораторная работа № 2

Разработка модели внутризаводской логистики

Задача смоделировать внутризаводскую логистику между складами заготовок, цехом сборки и складом готовой продукции. У цеха есть свой парк грузовиков, которые доставляют детали для сборки или продукцию на склады. Детали доставляются в цех сборки, если их запас в цехе стал менее 10 штук. Продукция вывозится из цеха раз в час.

Этап 1. Создание агентов модели

Создайте новую модель и назовите ее **Logistic** (рис. 2.1). Задайте единицы модельного времени — минуты.

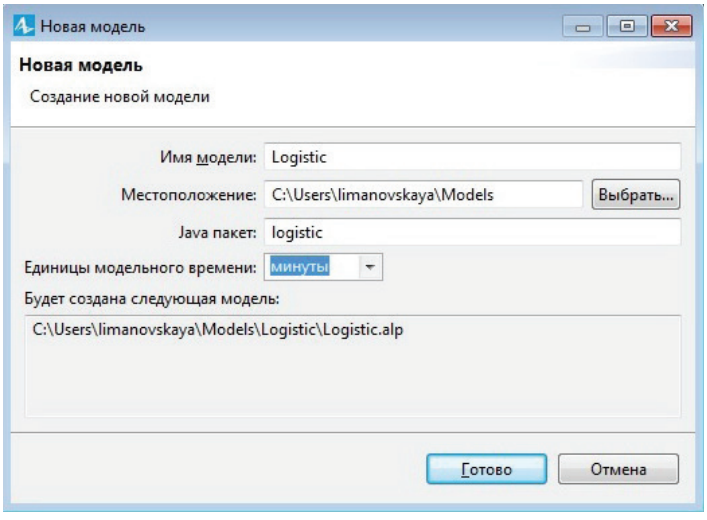

Рис. 2.1. Создание новой модели

В этой работе будет использован агентный подход. Все инструменты, необходимые для агентного подхода, находятся в библиотеке **Агент** на вкладке **Палитра** (рис. 2.2).

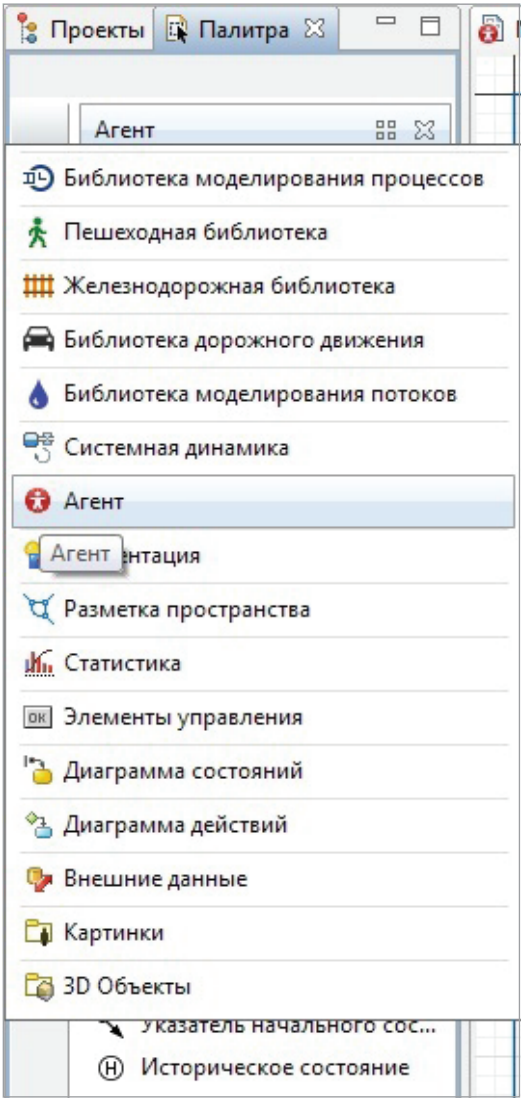

Рис. 2.2. Библиотека **Агент**

Задание агента Storage

Перейдите в библиотеку и перетащите компонент **Агент** на рабочее поле модели (рис. 2.3).

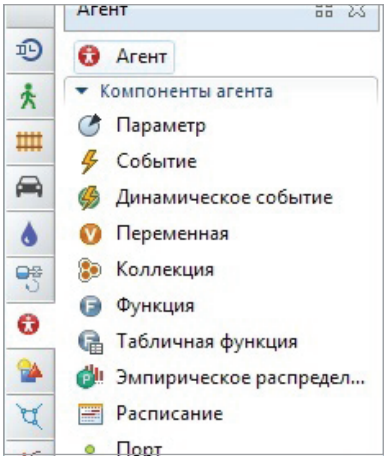

Рис. 2.3. Компонент **Агент**

Откроется мастер создания агентов. Поскольку все склады и цех будут в единственном экземпляре, то мы будем создавать **Единственного агента**. Выберите **Я создаю единственного агента** на первом шаге **Мастера** (рис. 2.4).

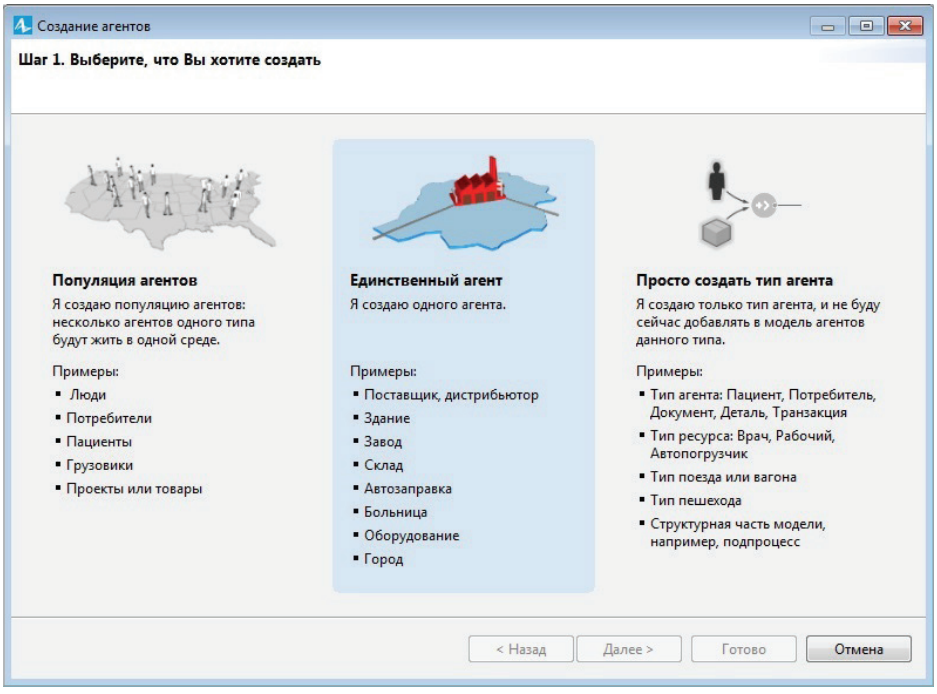

Рис. 2.4. Первый шаг мастера создания агента **Storage**

На втором шаге мастера задайте имя создаваемого агента **Storage** (рис. 2.5).

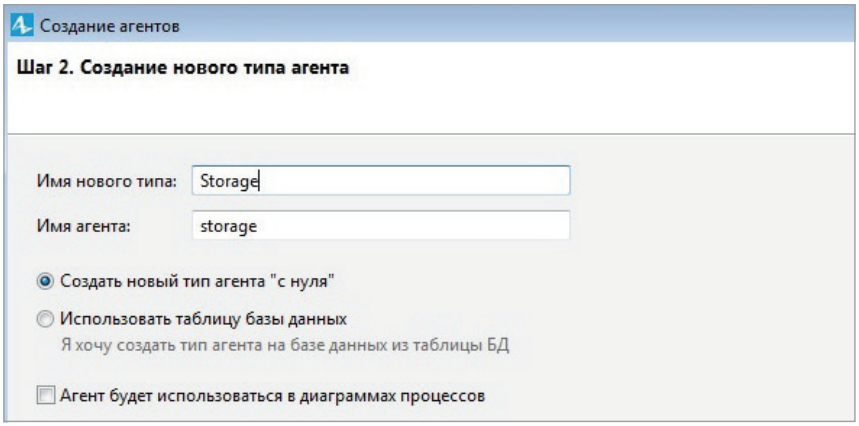

Рис. 2.5. Второй шаг мастера создания агента **Storage**

На третьем шаге мастера задайте анимацию агента (рис. 2.6).

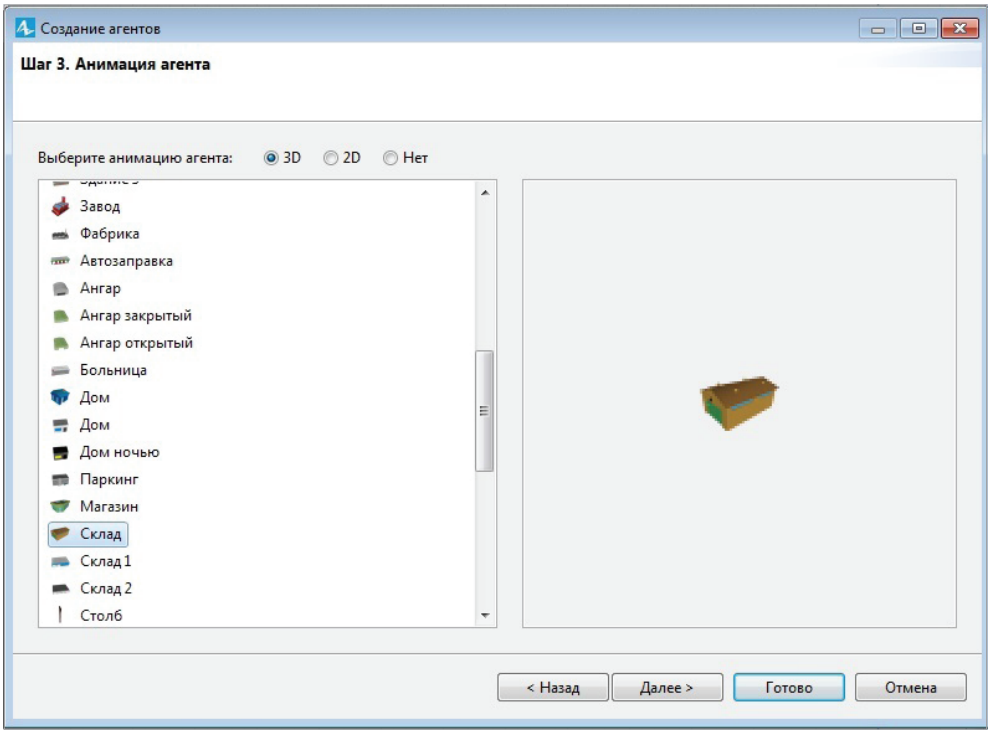

Рис. 2.6. Третий шаг мастера создания агента **Storage**

На следующем шаге мастера (рис. 2.7) задайте параметры агента, а именно переменную для хранения количества деталей на складе — **number_of_detal**. Поскольку детали измеряются целыми числами, то тип этой переменной **int**.

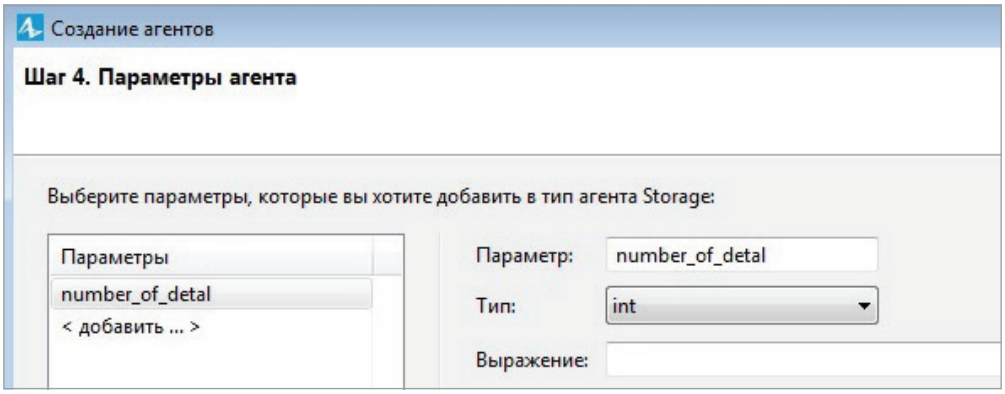

Рис. 2.7. Четвертый шаг мастера создания агента **Storage**

Нажмите кнопку **Готово** — на рабочем поле появится агент **Storage** (рис. 2.8).

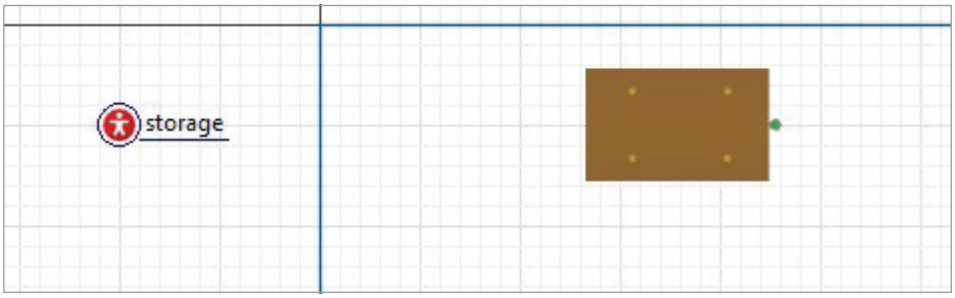

Рис. 2.8. Агент **Storage** на рабочем поле модели

Задание агента Storage_Production

Теперь создайте агента **Storage_Production**. Для этого перетащите элемент **Агент** на рабочее поле модели и пройдите шаги мастера с 1 по 4, задав имя агента (**Storage_Production**) и его анимацию (**Склад 1**). На втором шаге мастера будет предложен выбор между созданием нового типа агента и использованием существующего. Выберите новый тип агента. В результате на рабочем поле будут два агента **Storage** и **Storage_Production** (рис. 2.9).

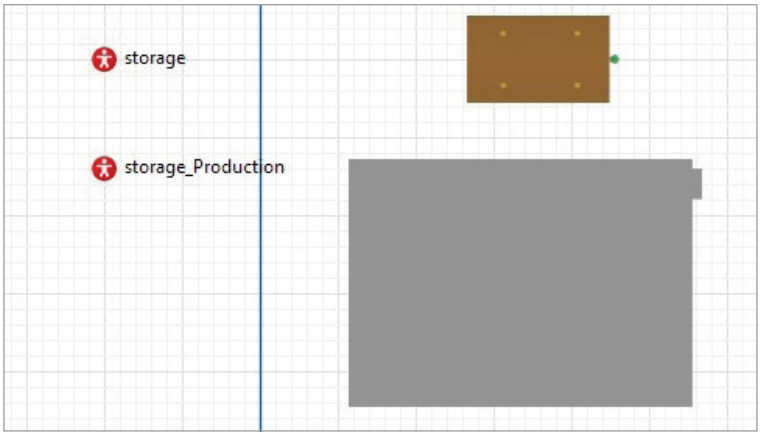

Рис. 2.9. Агенты **Storage** и **Storage_Production** на рабочем поле модели

Задание популяции агентов Truck

Теперь нужно создать множество агентов для моделирования грузовиков. Перетащите элемент **Агент** и на первом шаге мастера создания агента выберите **Популяция агентов** (рис. 2.10).

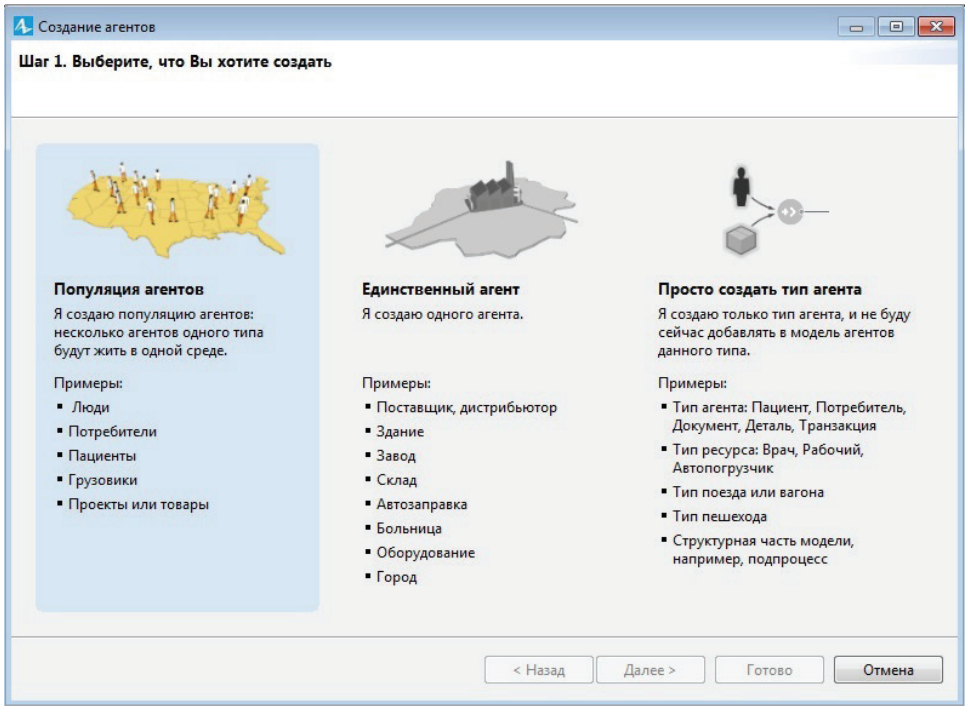

Рис. 2.10. Первый шаг мастера создания агентов **truck**

На втором шаге мастера выберите новый тип агента. На третьем шаге мастера задайте имя агента **Truck**. При этом одновременно будет создана популяция с именем **trucks**. На четвертом шаге выберите анимацию агента (рис. 2.11).

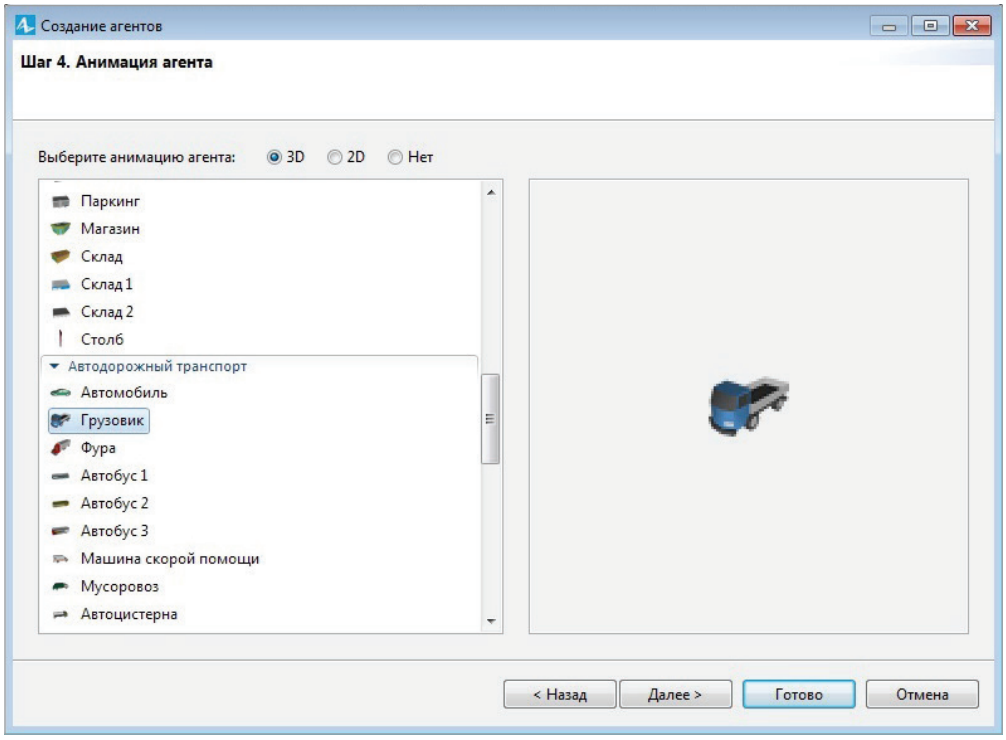

Рис. 2.11. Задание анимации агента **Truck**

На пятом шаге ничего не нужно задавать. На шестом шаге задайте объем популяции — 5 грузовиков. Нажмите кнопку Готово. Должно получиться три агента на рабочем поле (рис. 2.12).

Создание агента Plant

Создайте агент **Plant**. Для этого перетащите элемент **Агент** на рабочее поле модели и на первом шаге **Мастера** выберите **Создать единственного агента**. На втором шаге необходимо создать новый тип агента. На третьем шаге задайте имя агента **Plant**. На четвертом выберите анимацию (рис. 2.13).

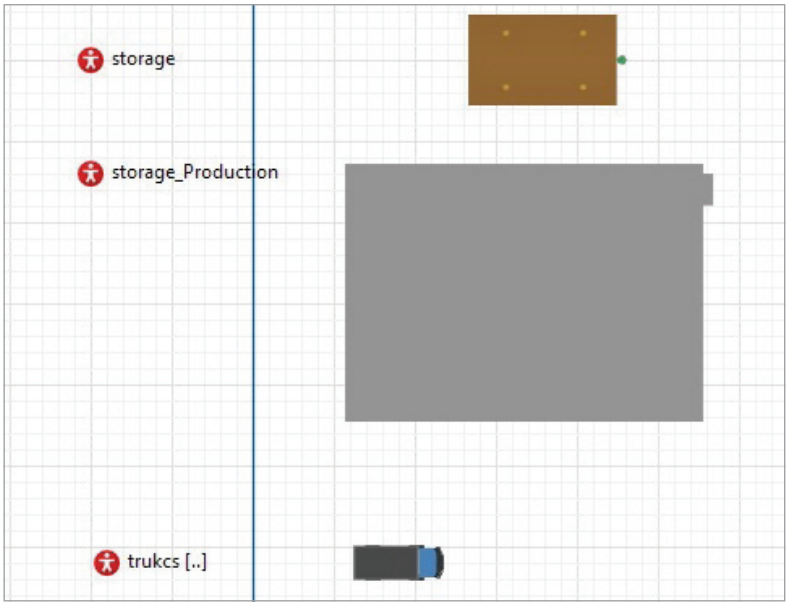

Рис. 2.12. Агенты на рабочем поле

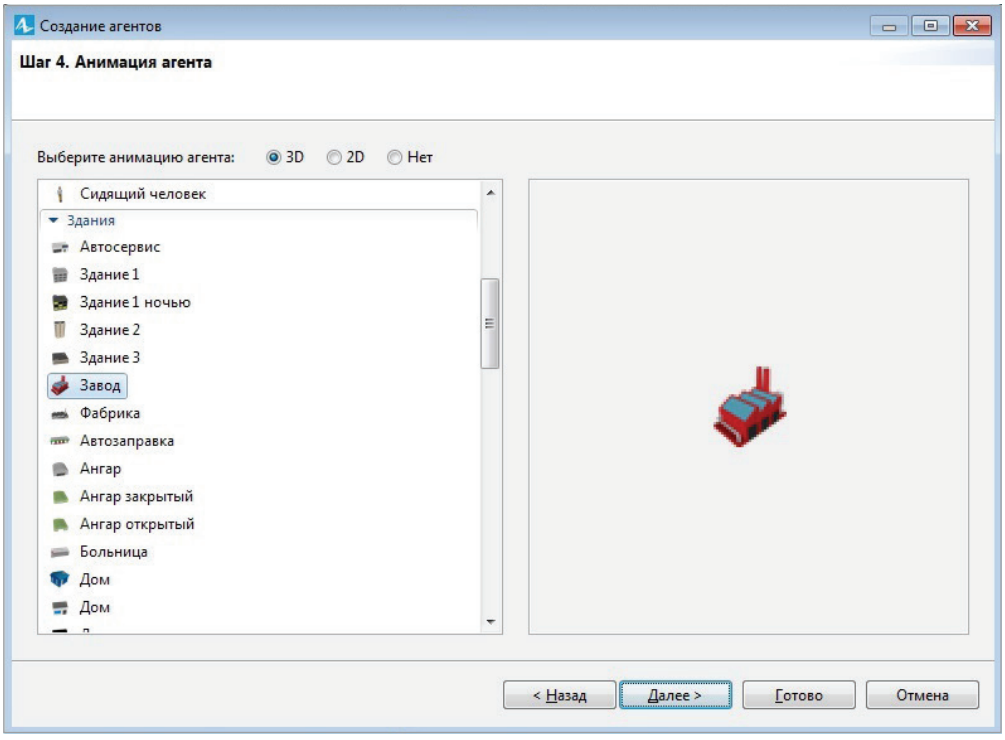

Рис. 2.13. Задание анимации агента **Plant**

На пятом шаге мастера создайте переменную, в которой будет храниться количество деталей в цехе — **number_of_detal_in_plant**, типа **int**.

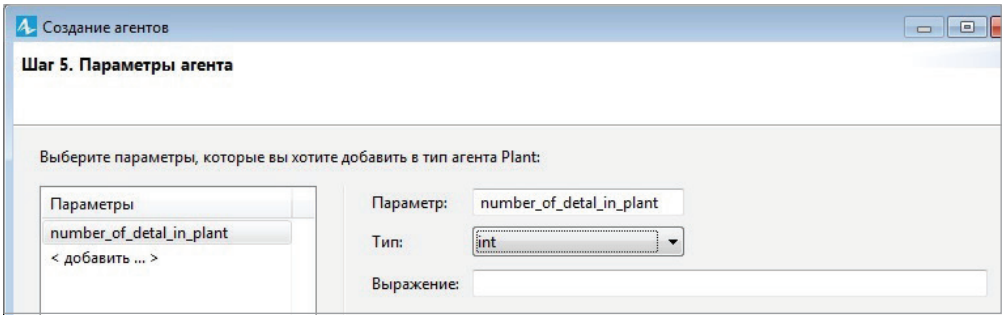

Рис. 2.14. Задание параметров агента **Plant**

Нажмите кнопку **Готово**. Должно быть так же, как на рис. 2.15.

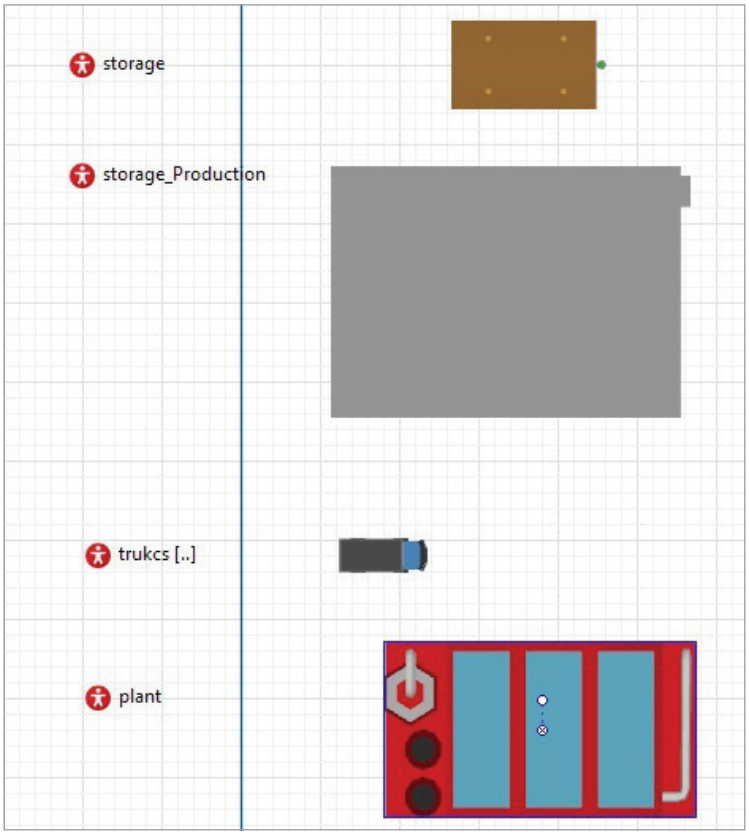

Рис. 2.15. Результаты этапа 1

Этап 2. Размещение агентов в пространстве модели

Агенты после их создания находятся в пространстве модели там, куда их поместили. Разместим их по координатам модельного пространства. Склад с деталями находится в точке с координатами X = 100, Y = 200. Склад готовой продукции — X = 1000, Y = 400. Сам цех — X = 300, Y = 200.

Для расстановки всех агентов по их координатам напишем функцию. Для создания функции используется элемент **Функция** (рис. 2.16).

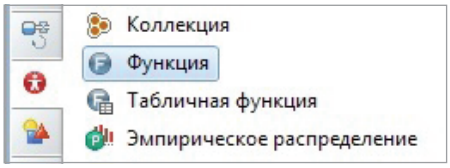

Рис. 2.16. Элемент **Функция**

Перетащите его на рабочее поле модели и перейдите в его свойства. В первом разделе свойств задайте имя функции — **Set**. Далее, поскольку эта функция не должна ничего считать, а должна просто расставить по координатам наших агентов, отметьте пункт **Действие** (рис. 2.17).

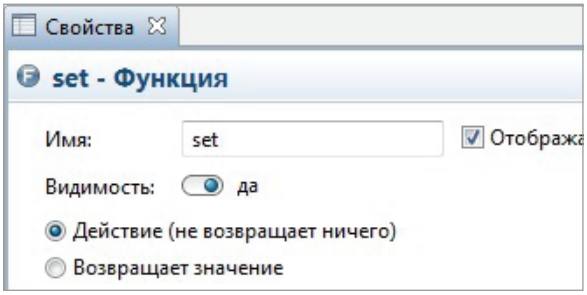

Рис. 2.17. Задание функции **Set**

Далее, нужно написать тело функции, т. е. ту программу, которую она будет выполнять. Для этого перейдите в раздел свойств **Тело функции** и наберите представленный на рис. 2.18 код.

В этом коде идет обращение к функции SetXY (X, Y) агентов **storage**, **storage_Production**, **plant** и каждого агента из популяции **trukcs**. Функция SetXY () ставит агента в указанные в аргументах функции координаты.

Тело функции

```
storage.setXY(100, 200);
storage_Production.setXY(1000,400);
plant.setXY(300,200);
for (int i=0;i<trukcs.size();i++)
{
    trukcs.get(i).setXY(300,200);
}
```

Рис. 2.18. Тело функции **Set**

Для того чтобы функция запустилась на выполнение, ее нужно вызвать. Организуем ее вызов при запуске главного агента **main**. Для этого откройте агент **main** (рис. 2.19) и щелкните на свободном в модели месте так, чтобы ни один из элементов агента **main** не был выделен.

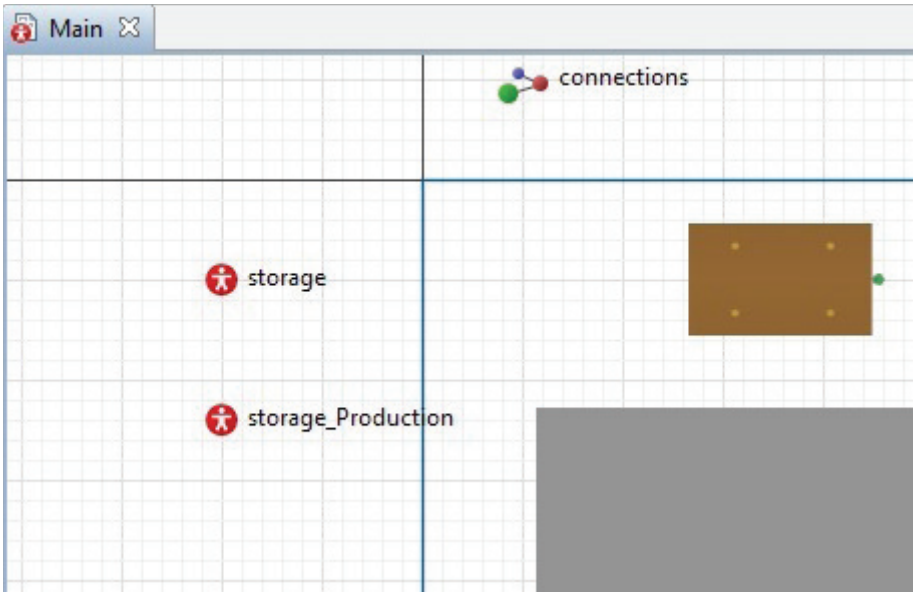

Рис. 2.19. Агент **main**

Перейдите в свойства агента **main** и в разделе **Действия агента** в пункте **При запуске** вызовите функцию **set ()** (рис. 2.20). Обратите внимание, что после **set ()** нужно поставить знак **;**.

Запустите модель на выполнение (рис. 2.21).

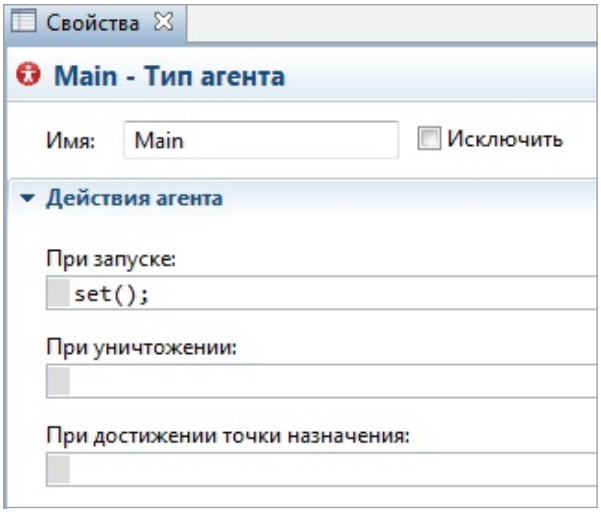

Рис. 2.20. Вызов функции **set ()**

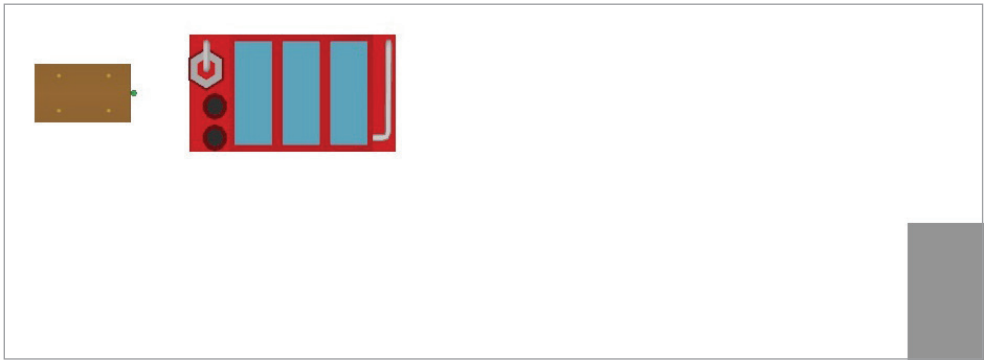

Рис. 2.21. Расположение агентов в модели

Этап 3. Моделирование производства деталей на складе

На складе производятся детали с интенсивностью 5 штук в час. Для моделирования их производства в упрощенном виде, т. е. будем просто моделировать увеличение количества деталей на 5 штук в час, перейдите в агент **storage**. В этом агенте уже сеть параметр **number_of_detal**, который содержит количество деталей на складе (рис. 2.22).

Увеличение количества деталей на складе зададим с помощью элемента **Событие**. Этот элемент запускает на выполнение либо заданную функцию, либо заданный в нем код.

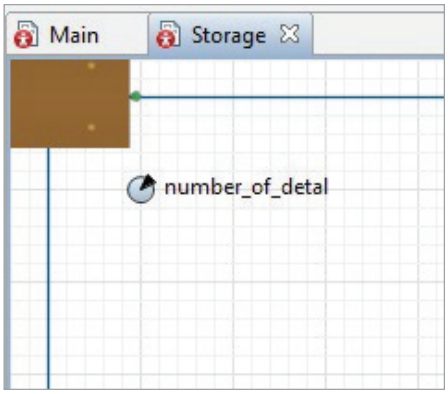

Рис. 2.22. Окно агента **storage**

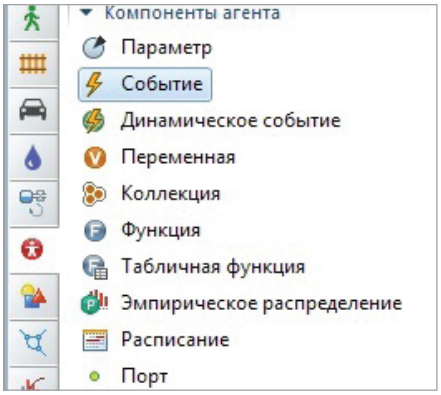

Рис. 2.23. Элемент **Событие**

Перетащите элемент **Событие** на рабочее поле агента **storage** и зайдите в свойства элемента **Событие** (рис. 2.24).

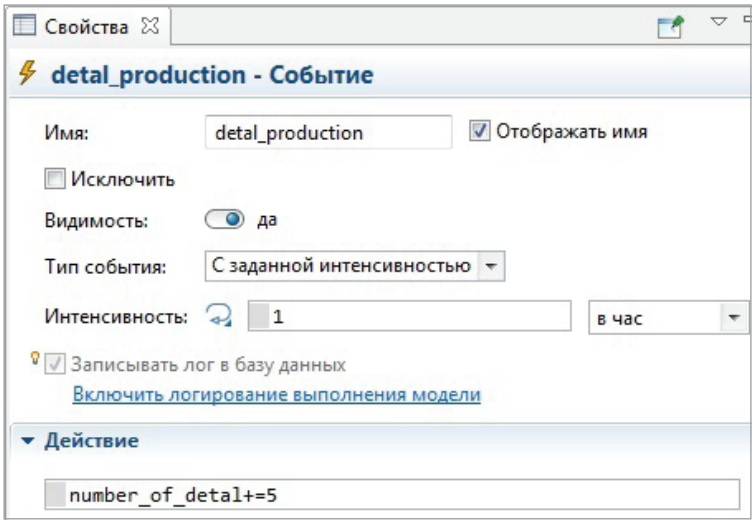

Рис. 2.24. Свойства элемента **detal_production**

Задайте имя события — **detal_production**, тип события — **С заданной интенсивностью** и **Интенсивность** 1 раз в час. Это значит, что событие будет выполняться раз в час. В разделе **Действие** задайте увеличение параметра **number_of_detal** на 5 штук. Значит, раз в час параметр **number_of_detal** будет увеличиваться на 5.

Запустите модель на выполнение и во время работы модели зайдите в агент **storage** (рис. 2.25).

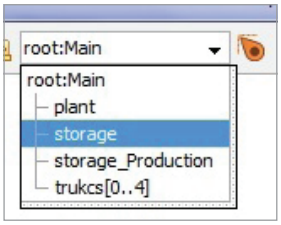

Рис. 2.25. Переход в агент **storage** во время выполнения моделирования

Рис. 2.26. Агент **storage** во время выполнения моделирования

Как видно из рис. 2.26, событие запускается и параметр **number_of_detal** растет.

Этап 4. Моделирование расхода деталей в цехе

Расход деталей в цехе промоделируем также с помощью элемента **Событие**.

Перейдите в агент **plant**. В нем уже есть параметр **number_of_detal_in_plant**. Добавим событие, которое будет уменьшать этот параметр на 4 штуки в час. Для этого перетащите элемент **Событие** на рабочее поле агента **plant** и задайте его свойства (рис. 2.27).

Запустите модель и зайдите в агент **plant**. Значение параметра **number_of_detal_in_plant** должно уменьшаться.

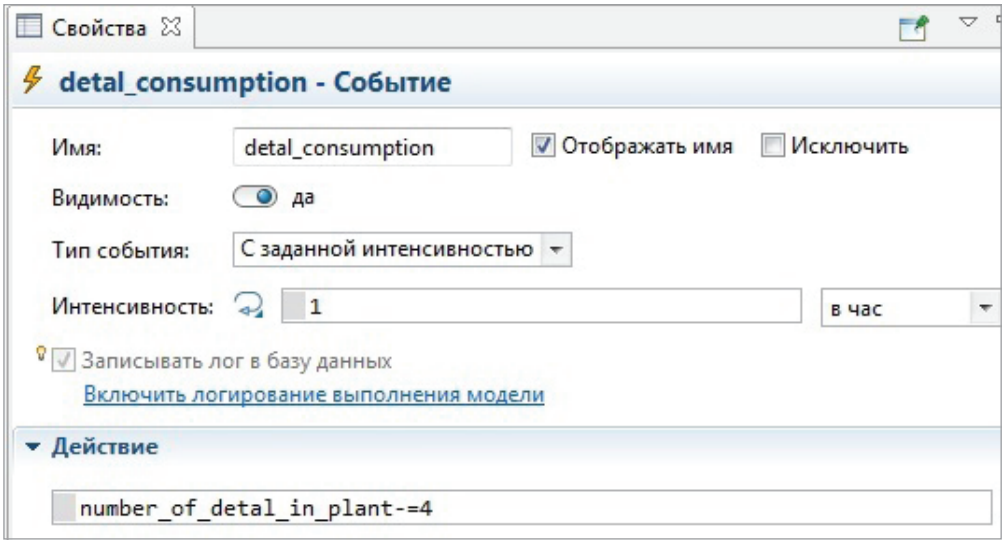

Рис. 2.27. Свойства события **detal_consumption**

Этап 5. Моделирование производства изделий в цехе

Для отображения количества произведенных в цехе изделий создадим параметр **number_of_izdeliya** в агенте **plant**. Для этого перетащите элемент **Параметр** на рабочее поле агента **plant** и задайте его свойства (имя параметра и его тип) так же, как на рис. 2.28.

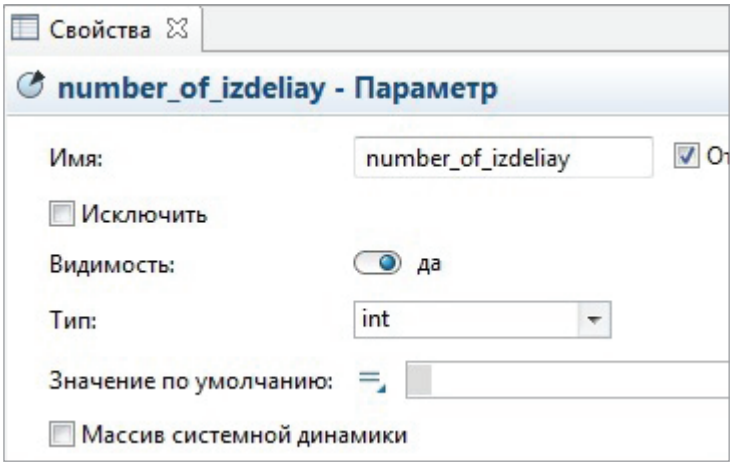

Рис. 2.28. Задание параметра **number_of_izdeliya**

Производство изделий смоделируем путем увеличения параметра **number_of_izdeliya** на 5 штук в час с помощью элемента **Событие**. Для этого перетащите элемент **Событие** на рабочее поле агента **plant** и задайте его свойства так же, как показано на рис. 2.29.

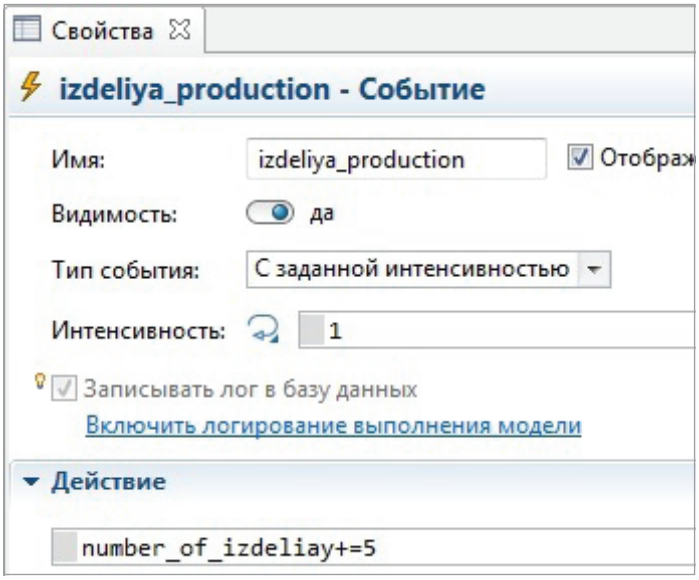

Рис. 2.29. Свойства события **izdeliay_production**

Запустите модель и перейдите в агент **plant**. Значение параметра **number_of_izdeliay** должно расти.

Этап 6. Моделирование движения грузовика с деталями от цеха к складу

Перейдите в агент **Truck**. Добавьте в него параметр **order** типа **int**. Для этого перетащите элемент **Параметр** на рабочее поле агента **Truck** и задайте его параметры так же, как на рис. 2.30.

Грузовик в модели может быть в трех состояниях: находиться в цехе, находиться на складе, ехать на склад. Для моделирования состояний грузовика используется диаграмма состояний. Все инструменты для ее построения находятся в библиотеке **Диаграмма состояний** вкладки **Палитра** (рис. 2.31).

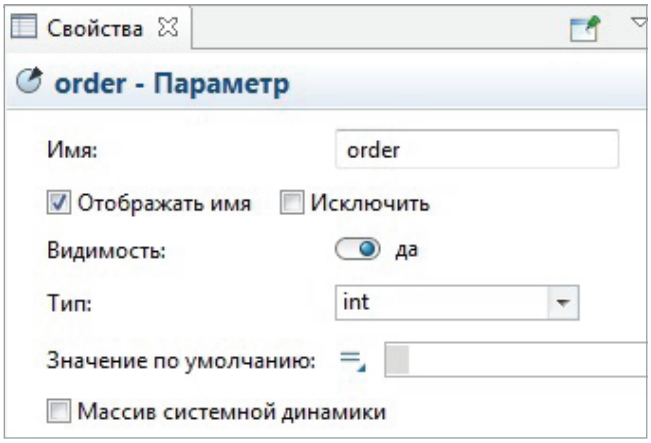

Рис. 2.30. Свойства параметра **order**

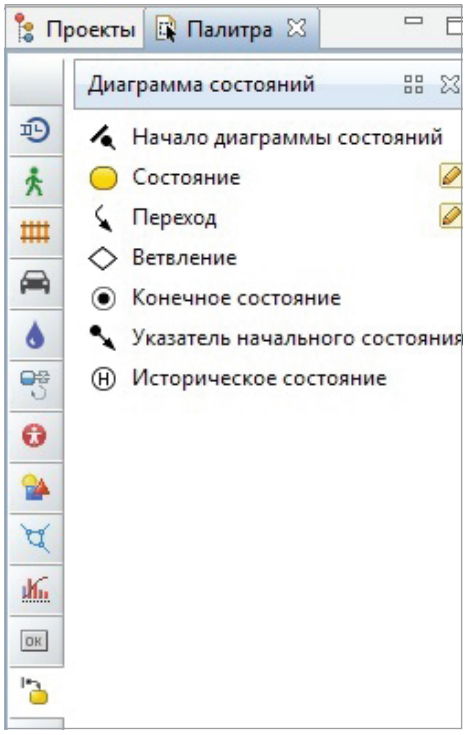

Рис. 2.31. Библиотека **Диаграмма состояний**

Перетащите на рабочее поле агента **Truck** элемент **Начало диаграммы состояний** и 3 элемента **Состояние**, дайте им имена и соедините их элементами **Переход** так же, как показано на рис. 2.32.

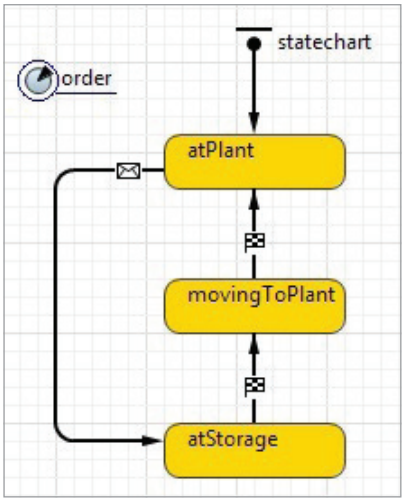

Рис. 2.32. Диаграмма состояний агента **Truck**

Зайдите в свойства перехода из состояния **atPlant** в состояние **atStorage** и задайте их, как показано на рис. 2.33.

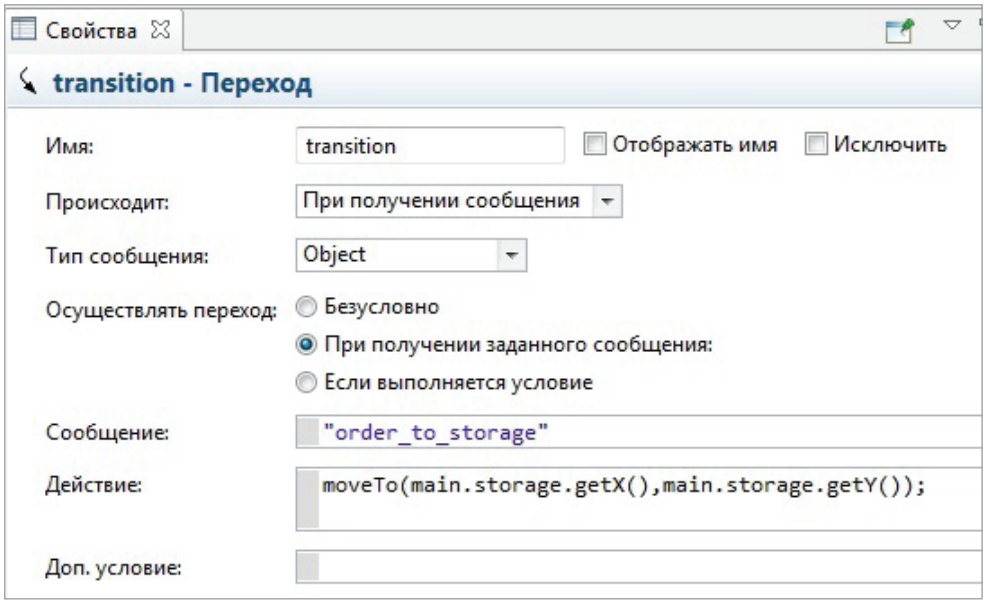

Рис. 2.33. Свойства перехода из состояния **atPlant** в состояние **atStorage**

Этот переход будет срабатывать, когда агент **Truck** получит сообщение **“order_to_storage”**. При срабатывании перехода агент **Truck** перей-

дет в состояние **atStorage** и выполнит действие, прописанное в разделе **Действие**, а именно поедет к месту склада. Задать сообщение, при котором будет срабатывать переход, можно любое.

Задайте свойства перехода из состояния **atStorage** в состояние **movingToPlant** так, как показано на рис. 2.34.

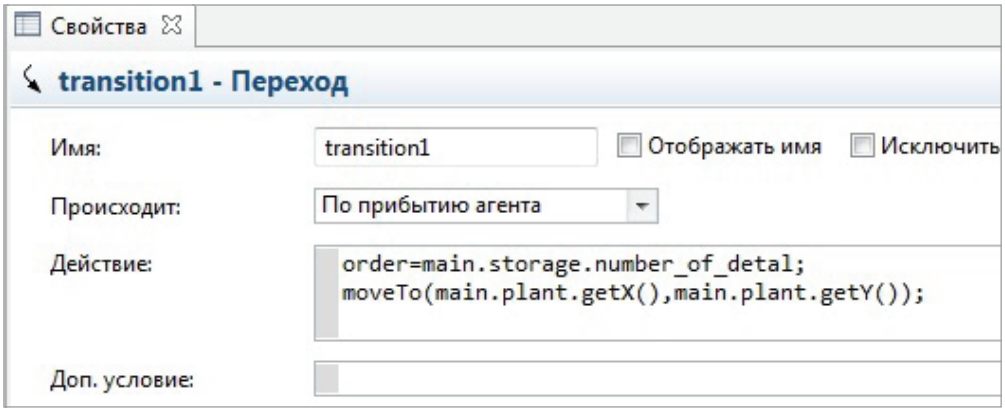

Рис. 2.34. Свойства перехода из состояния **atStorage** в состояние **movingToPlant**

Этот переход срабатывает сразу, как только агент прибывает на склад. При этом агент **Truck** переходит в состояние **movingToPlant** и выполняются действия, прописанные в разделе **Действие**, а именно параметр **order** становится равным количеству деталей на складе и грузовик уезжает в цех.

Задайте свойства перехода из состояния **movingToPlant** в состояние **atPlant** так, как показано на рис. 2.35.

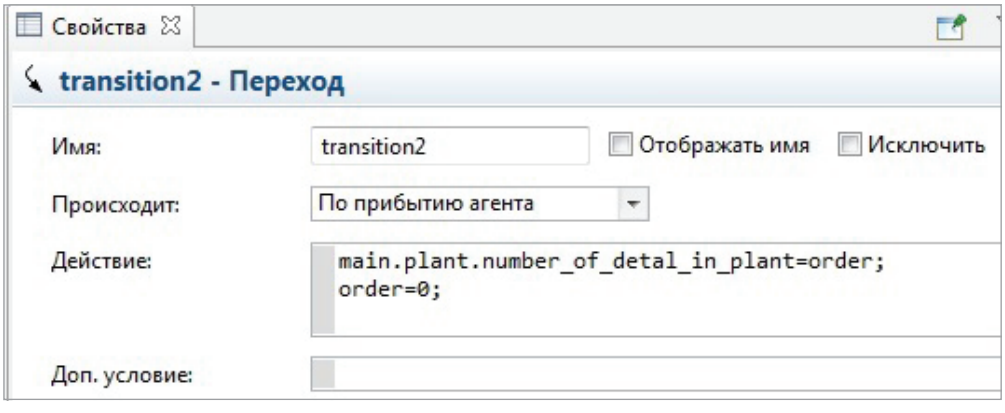

Рис. 2.35. Свойства перехода из состояния **movingToPlant** в состояние **atPlant**

Этот переход срабатывает сразу, как только агент начинает движение в цех. При этом агент **Truck** переходит в состояние **atPlant** и выполняются действия, прописанные в разделе **Действие**, а именно все содержимое параметра **order** переписывается в параметр **number_of_detal_in_plant** агента **Plant**, а сам параметр **order** обнуляется. Таким образом моделируется доставка деталей со склада в цех.

Задайте скорость движения грузовика 5 км в час, зайдя в свойства агента **Truck** (рис. 2.36).

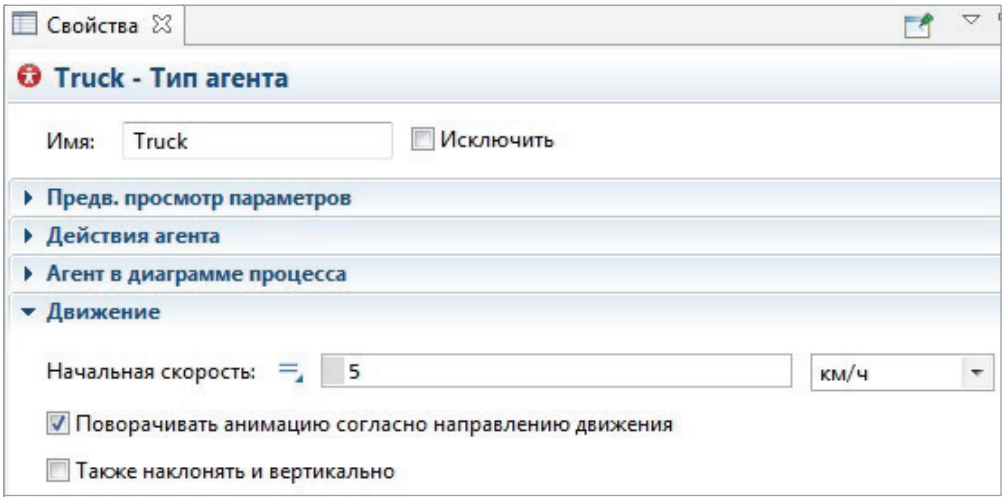

Рис. 2.36. Задание скорости движения грузовиков

Этап 7. Моделирование поиска свободного грузовика в цехе

Нам нужно найти в цехе свободный грузовик. Для этого перейдите в агент **Plant**.

Грузовик является свободным, если находится в цехе. В модели это состояние грузовика обозначено как **atPlant**. Для того чтобы найти свободный грузовик, нужно перебрать все грузовики коллекции **trucks** агента **Main** и найти такие, которые находятся в состоянии **atPlant**. Первый найденный и будет нужным нам грузовиком.

Ради разнообразия для написания функции поиска свободного грузовика воспользуемся инструментом **Диаграмма действий**. Такой инструмент находится в библиотеке **Диаграмма действий** панели **Палитра** (рис. 2.37).

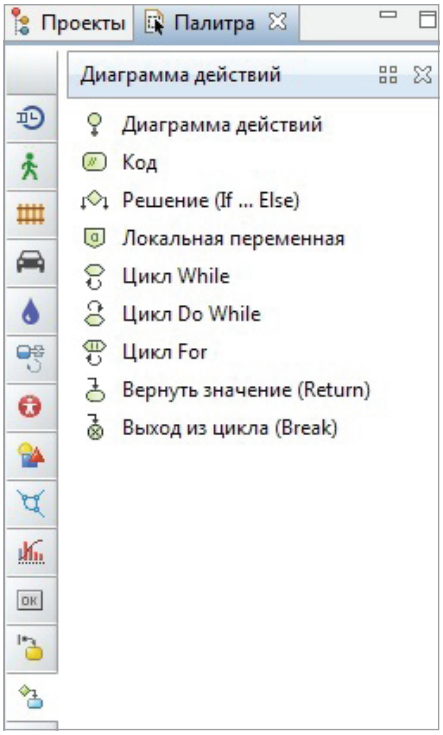

Рис. 2.37. Библиотека **Диаграмма действий**

Перетащите элемент **Диаграмма действий** на рабочее поле агента **Plant** (рис. 2.38).

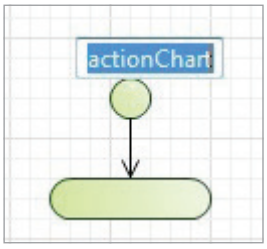

Рис. 2.38. Начало диаграммы действий

Перейдите в свойства диаграммы действий и задайте их, как показано на рис. 2.39. В свойствах задается имя функции **Find_truck** и тип возвращаемого значения — **Truck**. Это значит, что функция должна найти свободный грузовик, который в модели представлен агентом **Truck**, т. е. типом данных **Truck**. Если функция не найдет грузовик, то она вернет **Null**.

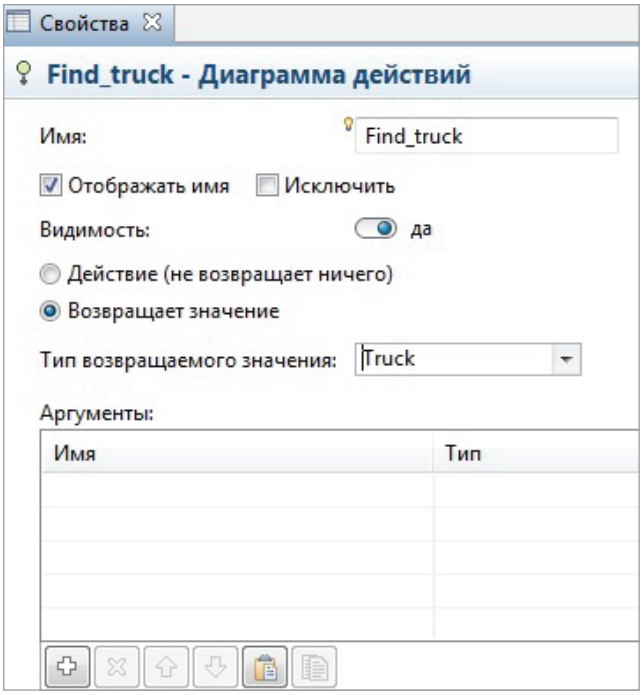

Рис. 2.39. Свойства функции **Find_truck**

Теперь перетащите элемент **Локальная переменная** так, чтобы появилась зеленая точка между началом и концом диаграммы (рис. 2.40). Должно получиться так, как показано на рис. 2.41.

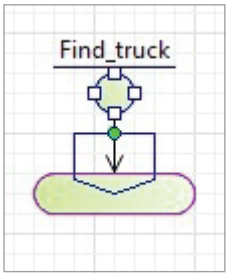

Рис. 2.40. Вставка элемента **Локальная переменная**

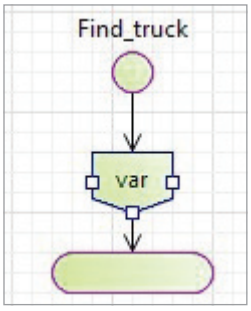

Рис. 2.41. Результат вставки элемента **Локальная переменная** в диаграмму действий

Задайте свойства элемента **Локальная переменная** так, как показано на рис. 2.42.

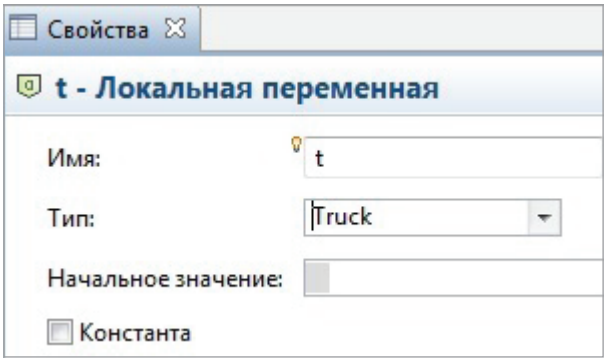

Рис. 2.42. Свойства элемента **Локальная переменная**

Таким образом была создана локальная переменная для сохранения найденного грузовика.

Поскольку надо организовать поиск по всей популяции грузовиков, то нужен цикл, проверяющий каждый грузовик. Перетащите элемент **Цикл For** и вставьте его между локальной переменной и концом диаграммы (рис. 2.43).

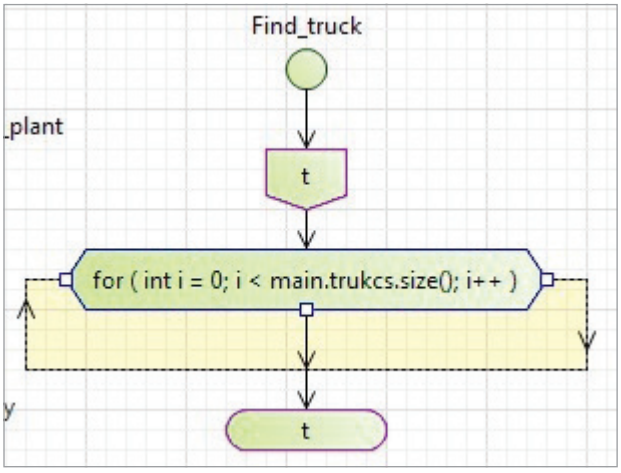

Рис. 2.43. Вставка цикла по коллекции

Задайте свойства цикла так, как показано на рис. 2.44.

Здесь задается начало и конец цикла. Цикл начинается с первого элемента коллекции и идет до последнего элемента.

Перетащите элемент **Код** и вставьте его между циклом и концом диаграммы так, как показано на рис. 2.45.

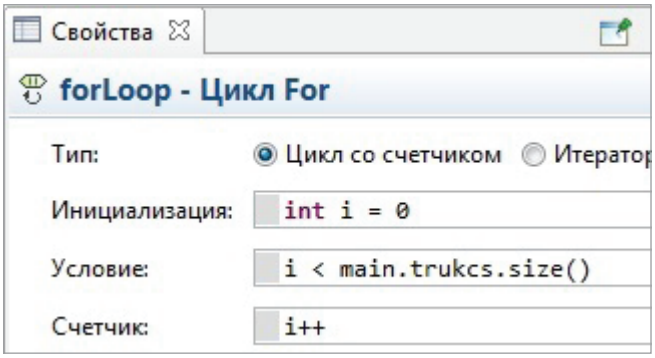

Рис. 2.44. Свойства цикла

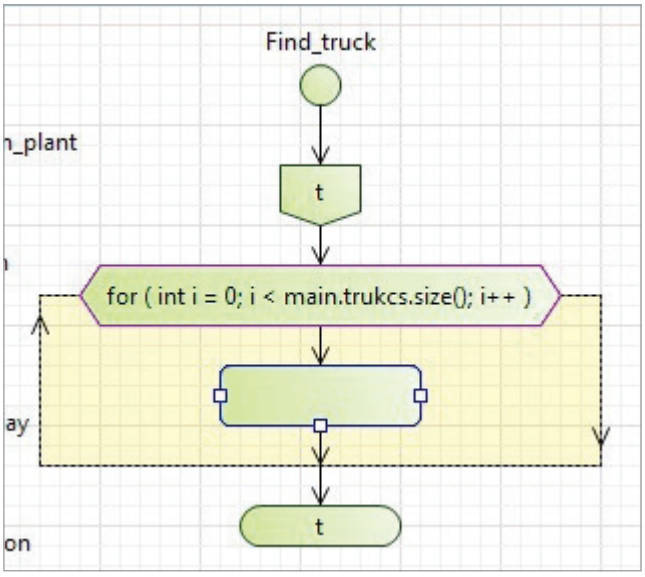

Рис. 2.45. Результат вставки элемента **Код**

Задайте свойства элемента **Код** так, как показано на рис. 2.46.

В свойствах в переменную **t** записывается очередной грузовик из коллекции.

Перетащите элемент **Решение** и вставьте его между элементами **Код** и концом диаграммы (рис. 2.47).

Задайте свойства элемента **Решение** так, как показано на рис. 2.48.

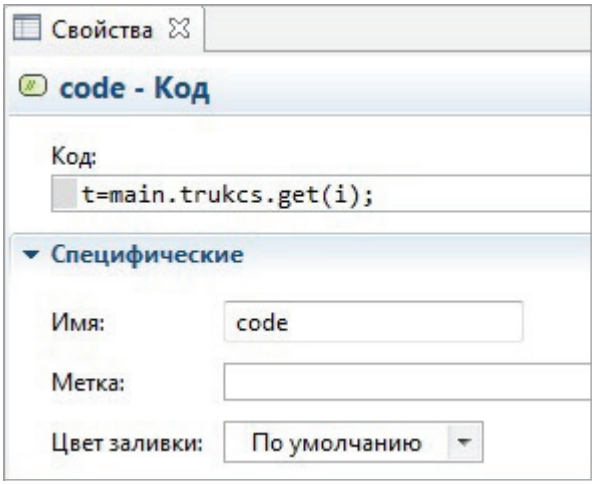

Рис. 2.46. Свойства элемента **Код**

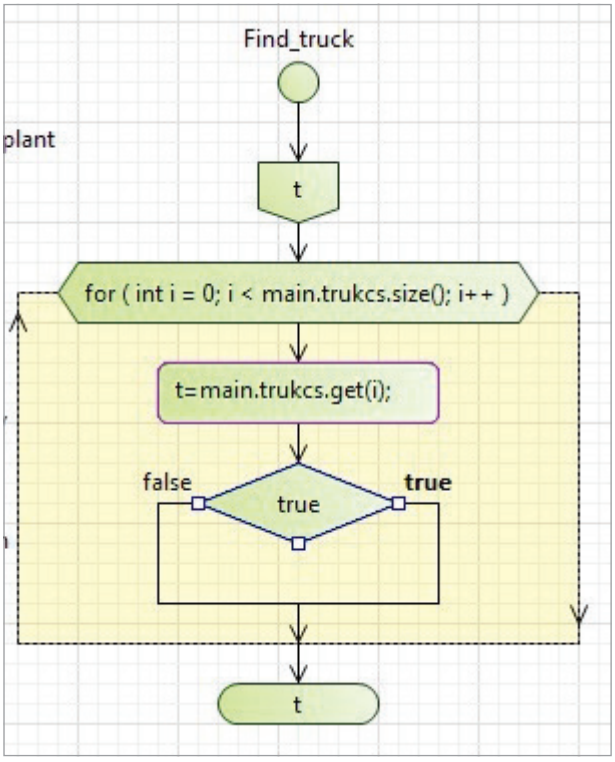

Рис. 2.47. Вставка элемента **Решение**

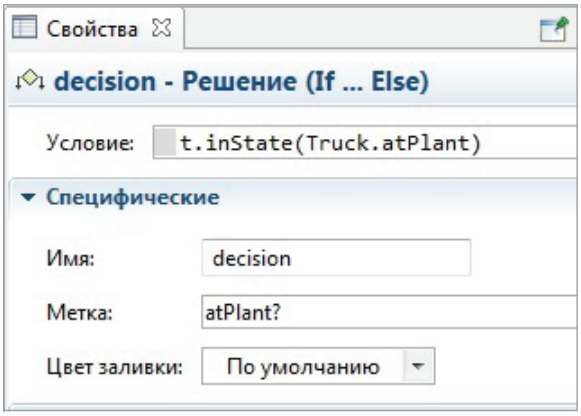

Рис. 2.48. Свойства элемента **Решение**

В свойствах задано условие проверки, а именно состояние элемента **t** (т. е. грузовика). Для красоты вводится метка, на которой написано само условие **atPlant**.

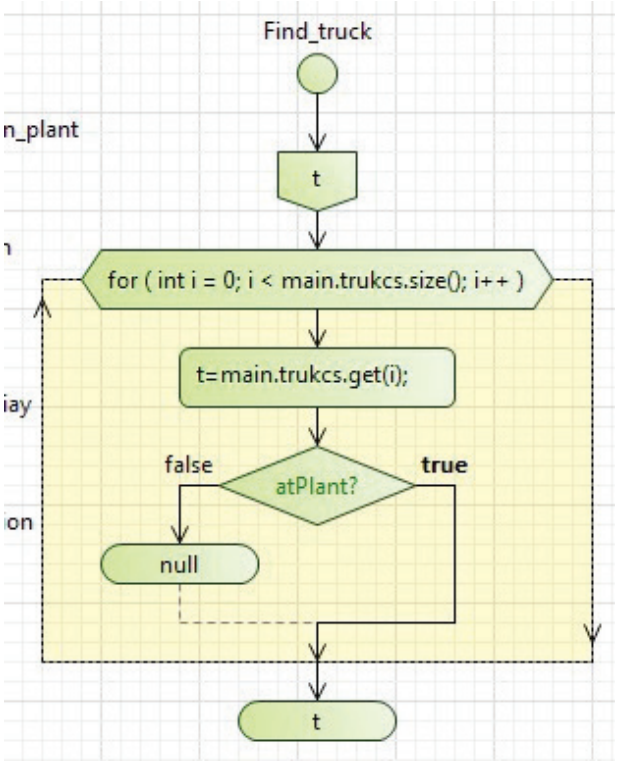

Рис. 2.49. Окончательный вид функции **Find_truck**

Перетащите элемент **Вернуть значение** в ветку **false** и введите в его свойствах **null**. В элемент **Вернуть значение** по ветке **true** введите **t** (рис. 2.49). Это означает, что если грузовик находится в состоянии **atPlant**, то функция вернет его значение; а если грузовик находится в любом другом состоянии, то функция вернет значение **null**.

Этап 8. Моделирование отправки грузовика из цеха за деталями на склад

Теперь нужно найденный свободный грузовик отправить на склад за деталями. Для этого используем функцию. В функции будет вызываться функция **Find_truck**. И, если грузовик будет найден и число деталей будет меньше 5, грузовику будет посылаться сообщение, по которому он стартует на склад.

Перетащите элемент **Функция** на рабочее поле агента **Plant** и задайте его свойства так, как показано на рис. 2.50.

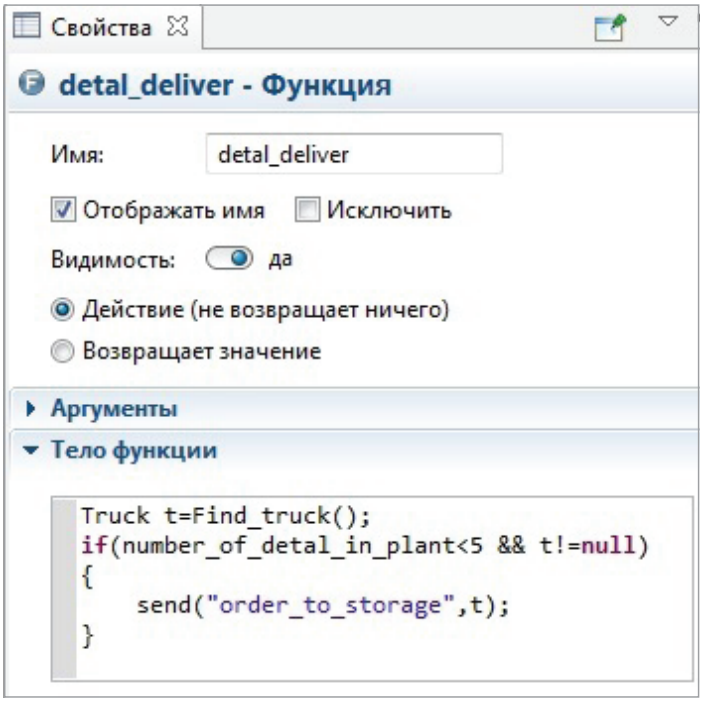

Рис. 2.50. Свойства функции **detal_deliver**

Для вызова функции используется событие, запускаемое каждую минуту. Перетащите элемент **Событие** и задайте его свойства так, как показано на рис. 2.51.

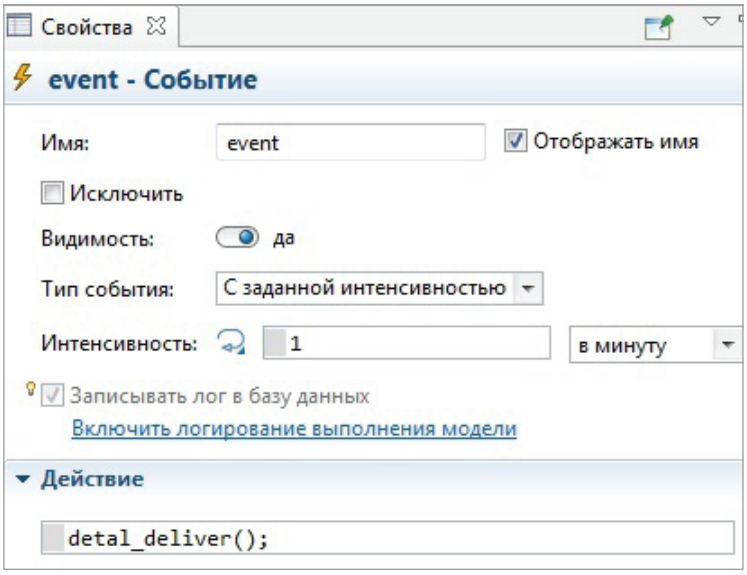

Рис. 2.51. Свойства события

Запустите модель. Грузовики должны ездить от склада к цеху и обратно. Количество деталей в цехе должно пополняться (рис. 2.52).

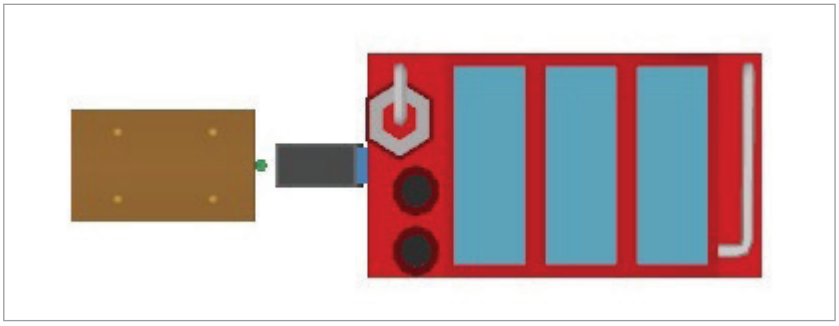

Рис. 2.52. Работа модели

Этап 9. Моделирование доставки готовой продукции из цеха на склад

Для доставки готовых изделий на склад готовой продукции используется грузовик большего объема, чем грузовики для доставки на склад деталей. Для него создадим отдельный агент **Lorry**.

Для этого перейдите в агент **main** и перетащите элемент **Агент** на рабочее поле модели. На первом шаге мастера выберите пункт **Создать единственного агента**. На втором шаге — создать новый тип агента. На третьем шаге задайте имя агента **Lorry**. На четвертом — выберите анимацию (фура). На пятом шаге создайте параметр агента **order** типа **int**. Перейдите в агента **Lorry**.

Грузовик в модели может быть в трех состояниях: находиться в цехе, находиться на складе готовой продукции и ехать на склад. Промоделируем все эти состояния с помощью диаграммы состояний. Постройте диаграмму состояний из трех состояний, просто перетаскивая элементы из библиотеки **Диаграмма состояний**. Соедините состояния переходами так, как показано на рис. 2.53.

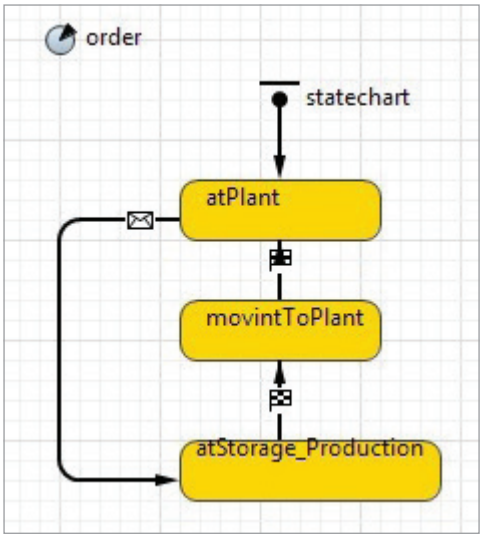

Рис. 2.53. Диаграмма состояний агента **Lorry**

Зайдите в свойства перехода из состояния **atPlant** в состояние **atStorage_Production** и задайте их так, как на рис. 2.54.

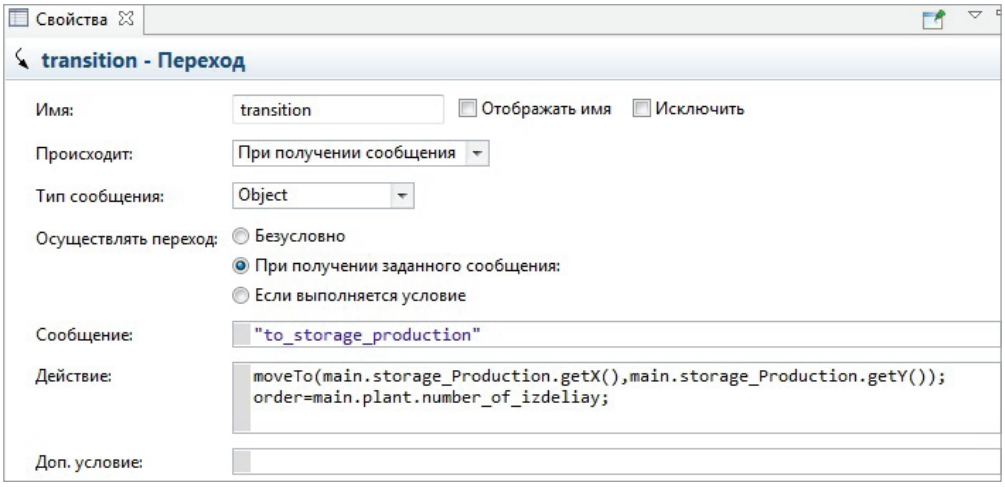

Рис. 2.54. Свойства перехода из состояния **atPlant** в состояние **atStorage_Production**

Такой переход будет происходить при получении сообщения **“to_storage_production”** агентом **Lorry**. При этом агент начнет движение к агенту **storage_Production** и в переменную **order** агента **Lorry** запишется количество произведенной в цехе продукции.

Выделите переход из состояния **atStorage_Production** в состояние **movingToPlant**. Перейдите в его свойства и задайте их так, как показано на рис. 2.55.

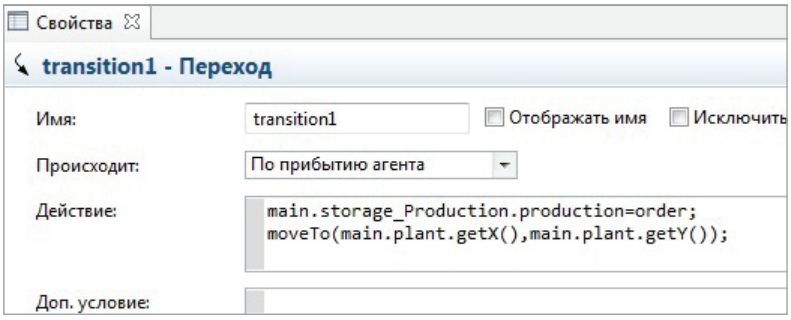

Рис. 2.55. Свойства перехода из состояния **atStorage_Production** в состояние **movingToPlant**

Такой переход будет происходить сразу после прибытия агента **Lorry** на склад готовой продукции, причем содержимое переменной **order** агента **Lorry** будет записываться в параметр **production** агента **storage_Production**. После этого грузовик отправляется к цеху.

Выделите переход из состояния **movingToPlant** в состояние **atPlant** и перейдите в его свойства. Задайте свойства перехода, как показано на рис. 2.56.

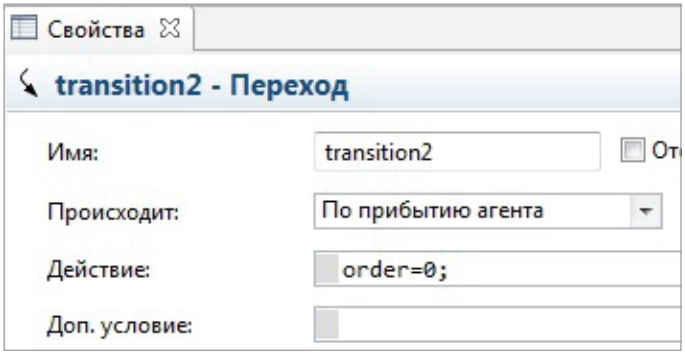

Рис. 2.56. Свойства перехода из состояния **movingToPlant** в состояние **atPlant**

Такой переход сработает, когда агент начнет движение в цех. При этом содержимое его параметра **order** обнулится.

Поскольку грузовик не всегда находится в цехе, в агенте **Plant** нужно организовать функцию для определения того, свободен ли грузовик в агенте **Plant**. Назовем ее **isFree** и построим с помощью **Диаграммы действий**. Перейдите в агент **Plant**. С помощью элементов библиотеки **Диаграмма действий** постройте функцию **isFree** так, как показано на рис. 2.57.

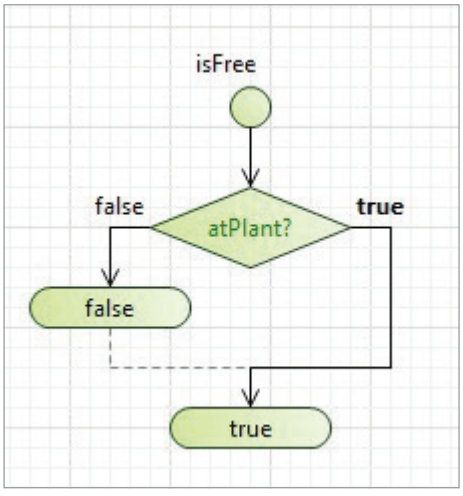

Рис. 2.57. Функция **isFree**

Выделите начальный элемент диаграммы и задайте свойства функции так, как показано на рис. 2.58.

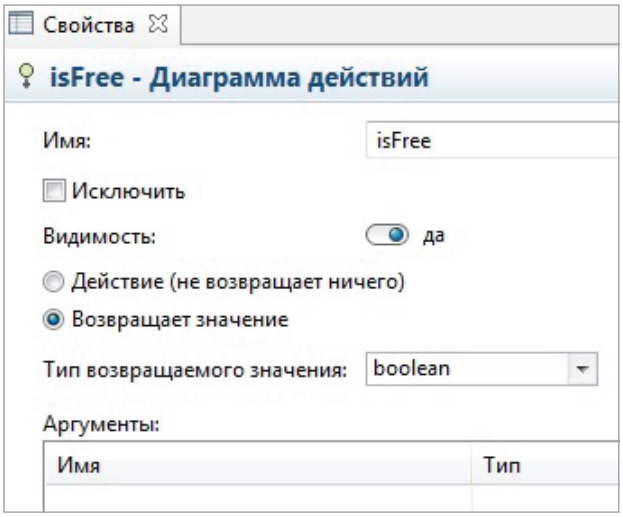

Рис. 2.58. Свойства функции **isFree**

Данная функция будет возвращать истину, если грузовик свободен, и ложь, если он занят.

Выделите элемент **Решение** и задайте его свойства так, как показано на рис. 2.59.

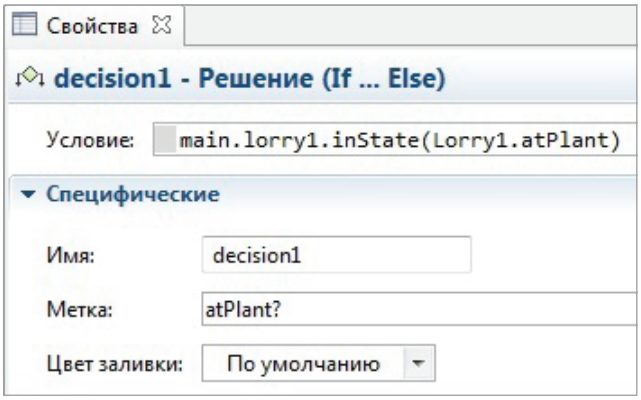

Рис. 2.59. Свойства элемента **Решение**

В элементе **Решение** проверяется информация о нахождении грузовика (агент **Lorry**) в состоянии **atPlant**, т. е. он находится в цехе. Если грузовик в цехе, то он свободен.

Отправкой грузовика на склад готовой продукции будет заниматься функция **izdeliya** агента **Plant**. Перетащите элемент **Функция** из агентной библиотеки и задайте его свойства так, как показано на рис. 2.60.

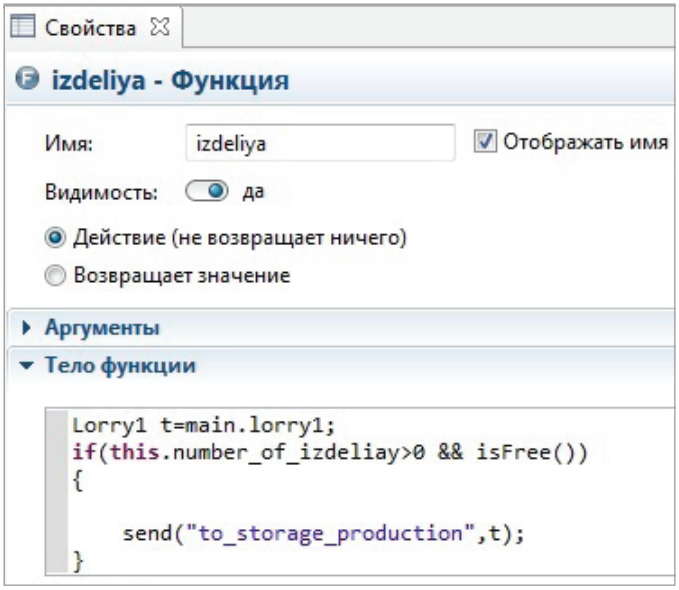

Рис. 2.60. Свойства функции **izdeliya**

В этой функции грузовик записывается в виде переменной **t**. Далее, если в цехе есть готовая продукция (параметр **number_of_izdeliya>0**) и есть свободный грузовик (функция **isFree ()** возвращает истину), то грузовику посылается сообщение **“to_storage_production”** и он отправляется на склад готовой продукции.

Для активации этой функции используется событие **izdeliya_deliver**. Перетащите элемент **Событие** из агентной библиотеки на рабочее поле агента **Plant** и задайте его свойства так, как показано на рис. 2.61.

Такое событие будет происходить 10 раз за час, и каждый раз будет вызываться функция **izdeliya**.

Запустите модель. От цеха должны двигаться грузовики на склад деталей и на склад готовой продукции (рис. 2.62).

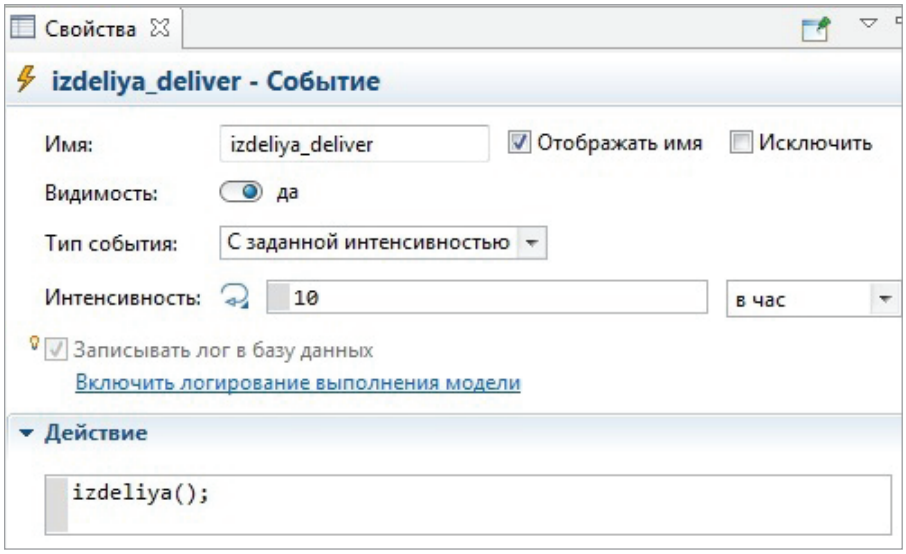

Рис. 2.61. Свойства события izdeliya_deliver

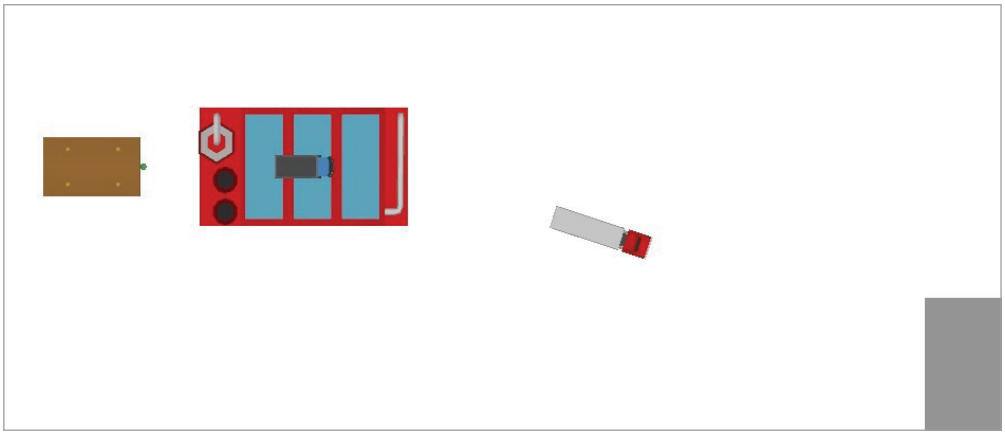

Рис. 2.62. Работа конечной модели

Лабораторная работа № 3

Использование анимации в дискретно-событийном подходе в AnyLogic 8.1

Задача

В этой работе нужно дополнить созданную в лабораторной работе № 1 модель технологической сборки изделия анимацией процесса.

Решение

Этап 1. Задание анимации агентов модели

В модели все должно быть прекрасно — и реализация, и анимация. Поэтому начнем с добавления к агентам модели анимации. При введении анимации процесса сам процесс конкретизируется и теперь нужно задать процесс сборки вполне конкретного изделия. Пусть это будет сборка настенных зеркал, т. е. процесс вставки зеркала в раму. Причем рама (первая деталь) подвергается сначала полировке, потом окраске. Заготовкой для зеркала служит стекло (вторая деталь), на которое наносят покрытие. В конце процесса зеркала упаковываются по 5 штук в коробку.

Перейдите в агент **Detal1**. Инструменты для рисования графических примитивов находятся в библиотеке **Презентация** вкладки **Палитра** (рис. 3.1).

Перетащите из библиотеки объект **Прямоугольник** на рабочее поле агента **Detal1** (рис. 3.2) и задайте его свойства (размеры и цвет) так, как показано на рис. 3.3.

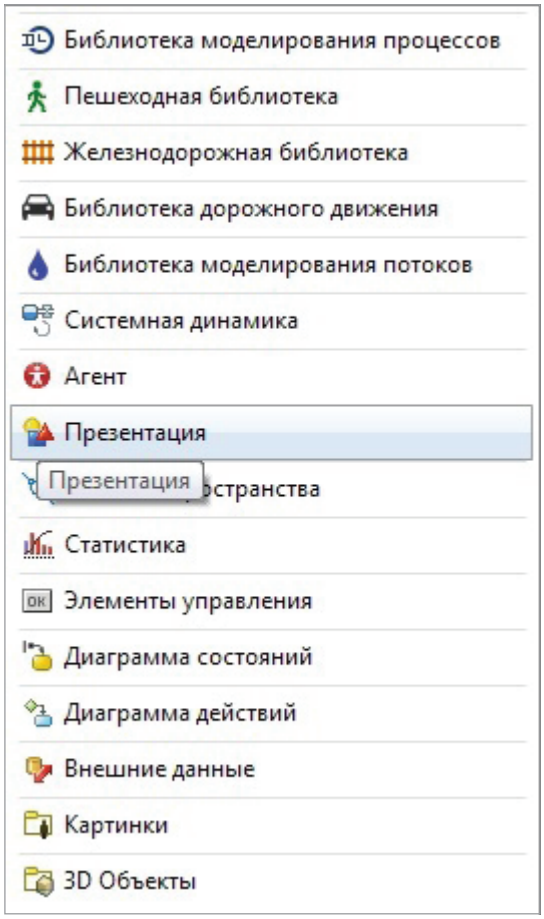

Рис. 3.1. Библиотека **Презентация**

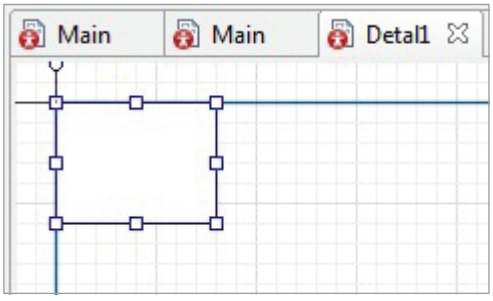

Рис. 3.2. Размещение объекта **Прямоугольник** в агенте **Detal1**

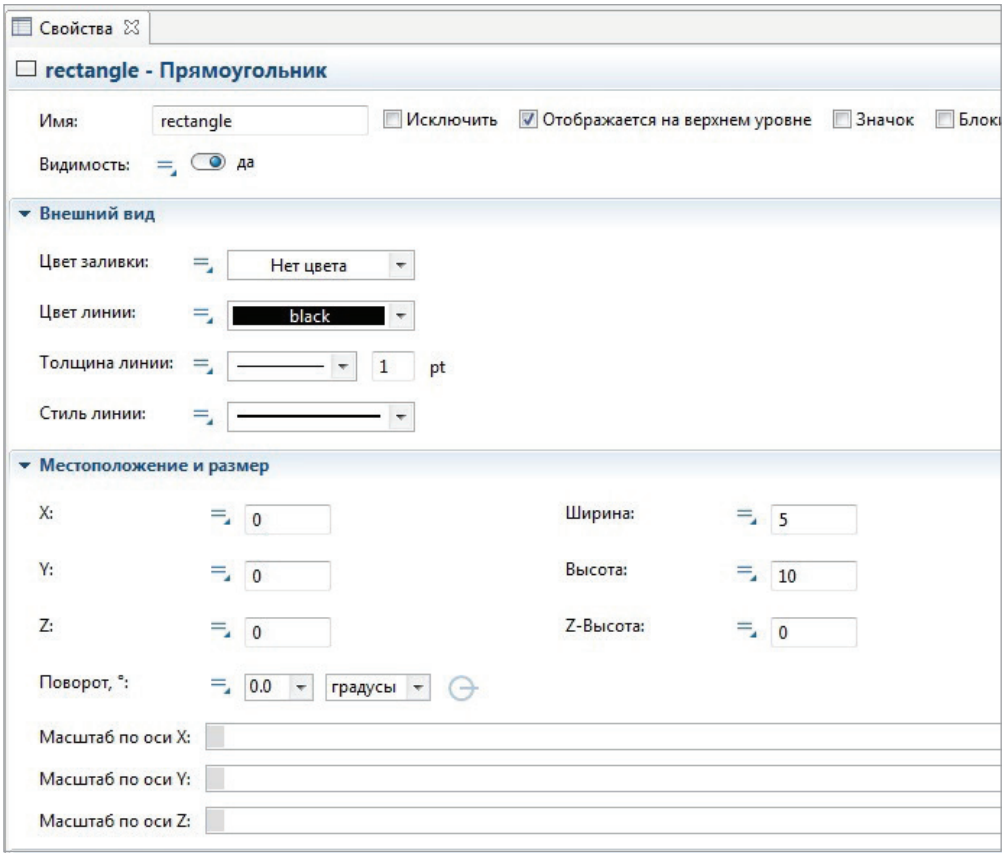

Рис. 3.3. Свойства объекта **Прямоугольник**

Перейдите в агент **Detal2** и постройте анимацию для заготовки под зеркало — прямоугольник тех же размеров, что и в агенте **Detal1**, но с другими свойствами (рис. 3.4).

Теперь задайте анимацию для готового зеркала в раме. Перейдите в агент **Izdelie** и создайте такой же прямоугольник и задайте его свойства, как показано на рис. 3.5.

Теперь зададим анимацию агента **Box**. Для его анимации используем стандартное изображение **Box**. Наборы стандартных картинок находятся в библиотеке **Картинки** вкладки **Палитра** (рис. 3.6).

Перейдите в агент **Box**, перетащите из библиотеки **Картинки** элемент **Коробка** (рис. 3.7) и задайте его координаты.

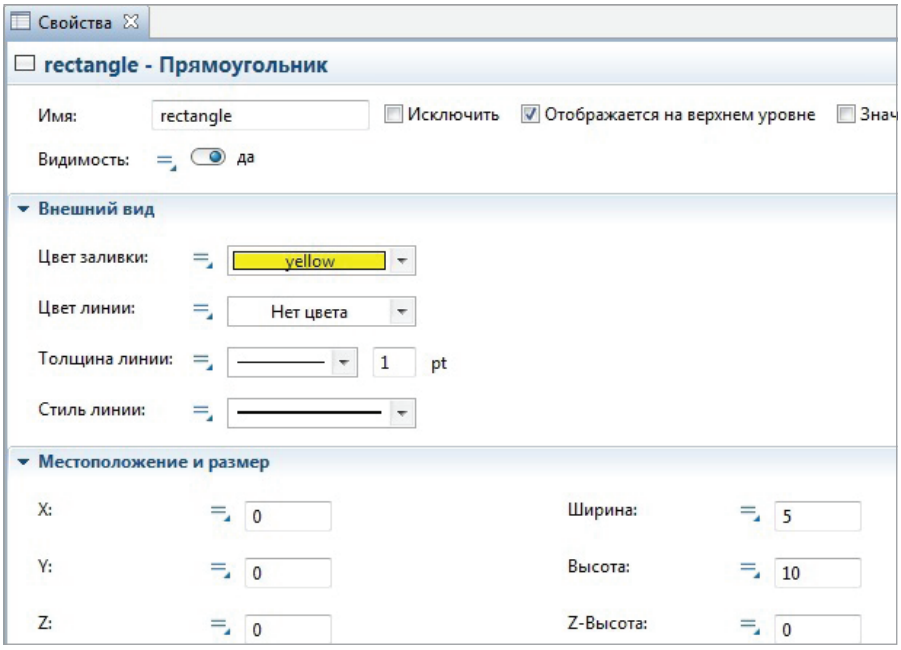

Рис. 3.4. Свойства объекта **Прямоугольник** агента **Detal2**

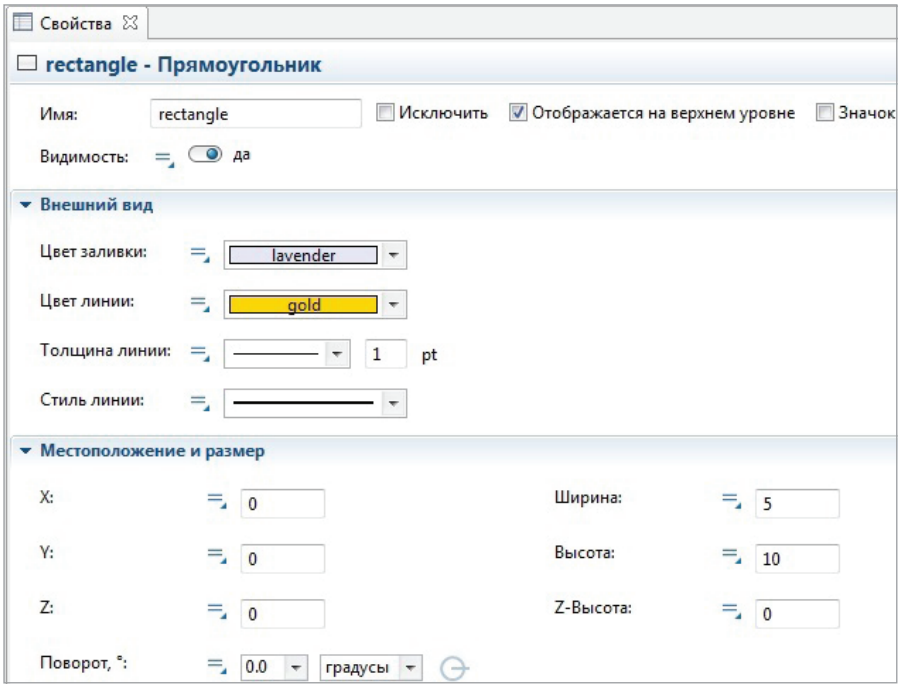

Рис. 3.5. Свойства анимации агента **Izdelia**

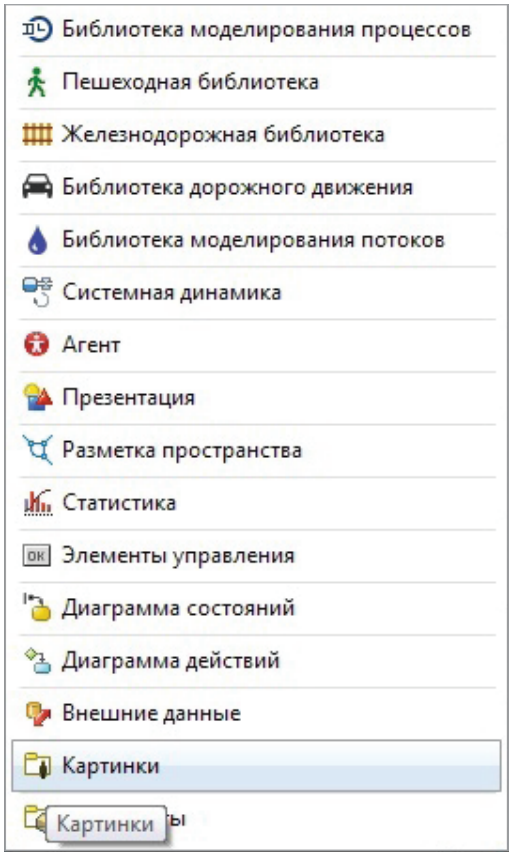

Рис. 3.6. Библиотека **Картинки**

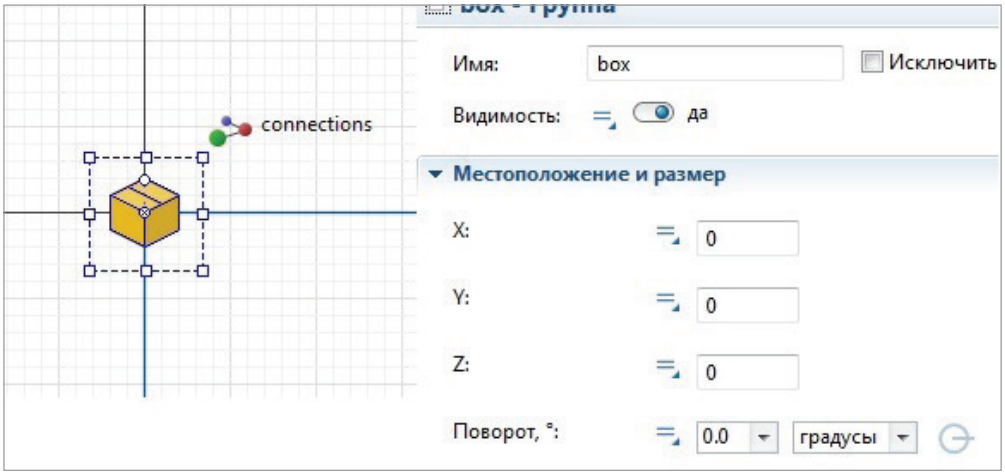

Рис. 3.7. Анимация агента **Box**

Этап 2. Задание чертежа сборочного цеха

В качестве фоновой картины модели в среде **AnyLogic** можно использовать любой графический файл, в том числе и чертежи объектов.

Для вставки графических объектов используется элемент **Изображение** из библиотеки **Презентация** вкладки **Палитра** (рис. 3.8).

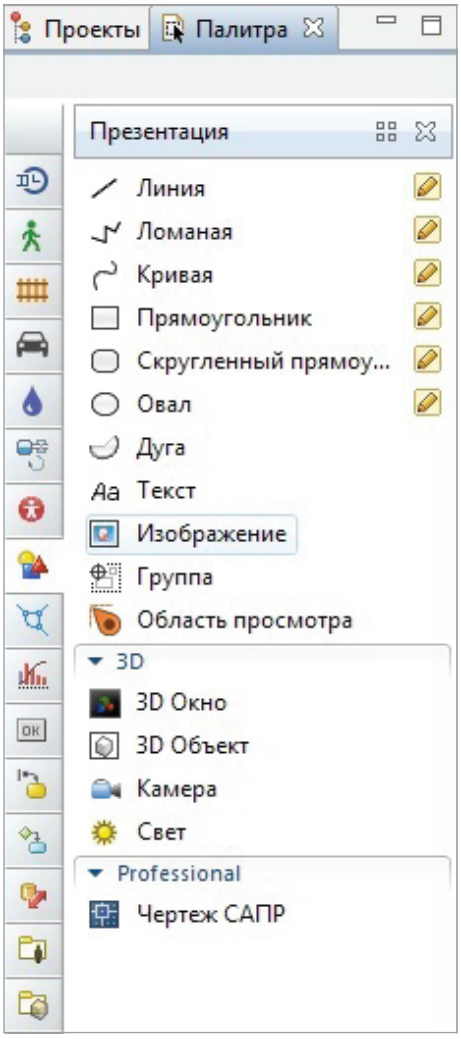

Рис. 3.8. Элемент **Изображение**

Перетащите элемент **Изображение** на рабочее поле агента **main** и в открывшемся окне выберите файл **рисЦеха.jpeg** (рис. 3.9).

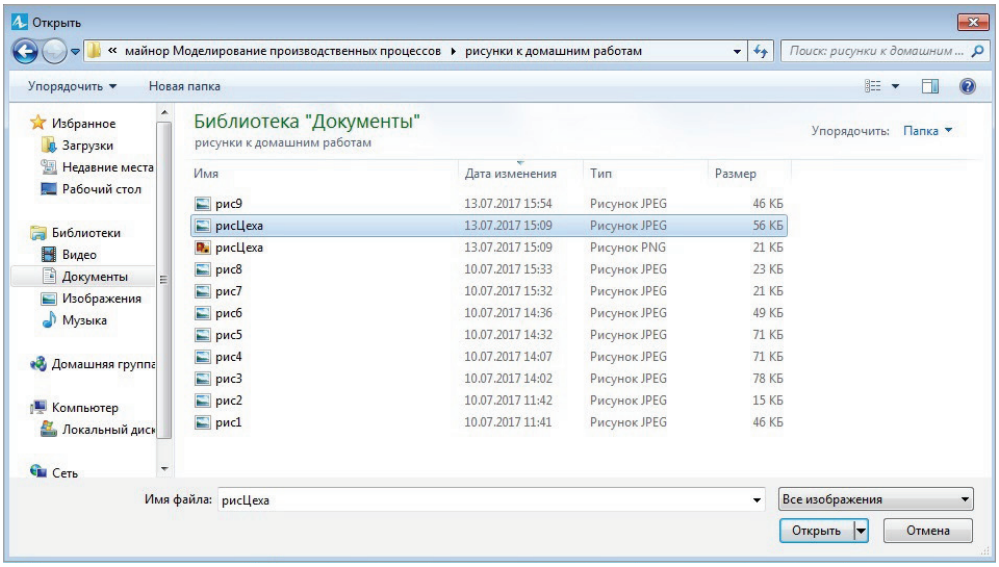

Рис. 3.9. Вставка чертежа цеха

После вставки должно получиться так, как показано на рис. 3.10.

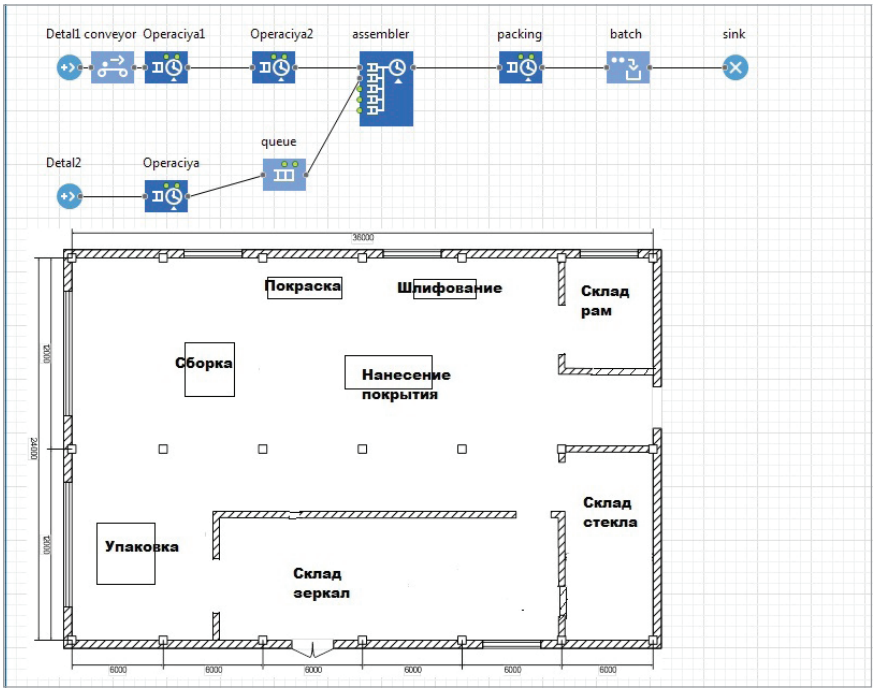

Рис. 3.10. Результат вставки чертежа склада

Этап 3. Задание места хранения заготовок для зеркал

Для полной анимации модели необходимо не только задать ее логику, но и привязать логические элементы модели к местам расположения реальных операций, мест хранения и т. д. Такая привязка выполняется с помощью инструментов библиотеки **Разметка пространства** (рис. 3.11).

Заготовки для зеркал (рамы и стекло) хранятся на двух разных складах. Для привязки мест их хранения к элементу, который моделирует их появление в модели (**source**), используется элемент **Прямоугольный узел** из библиотеки **Разметка пространства** (рис. 3.12). Щелкните на нем дважды и нарисуйте два таких элемента на чертеже цеха в местах склада рам и склада стекла, как показано на рис. 3.13.

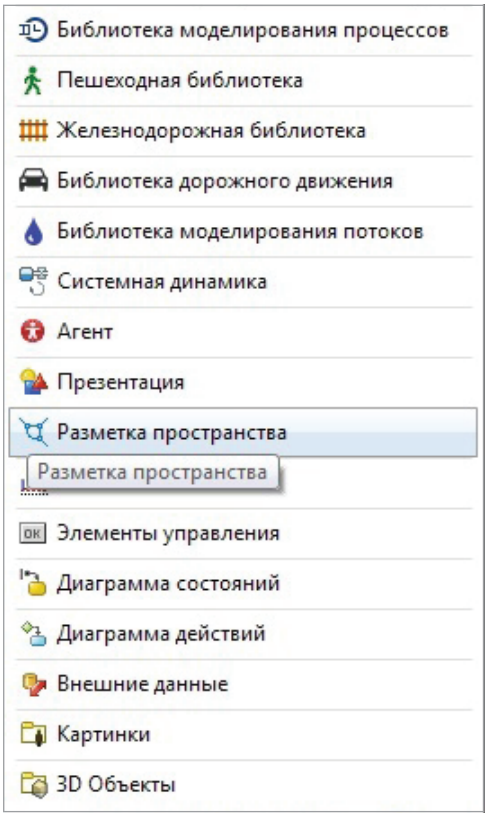

Рис. 3.11. Библиотека **Разметка пространства**

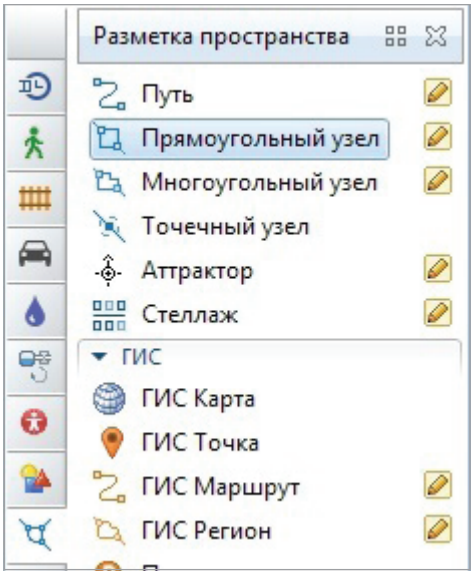

Рис. 3.12. Элемент **Прямоугольный узел**

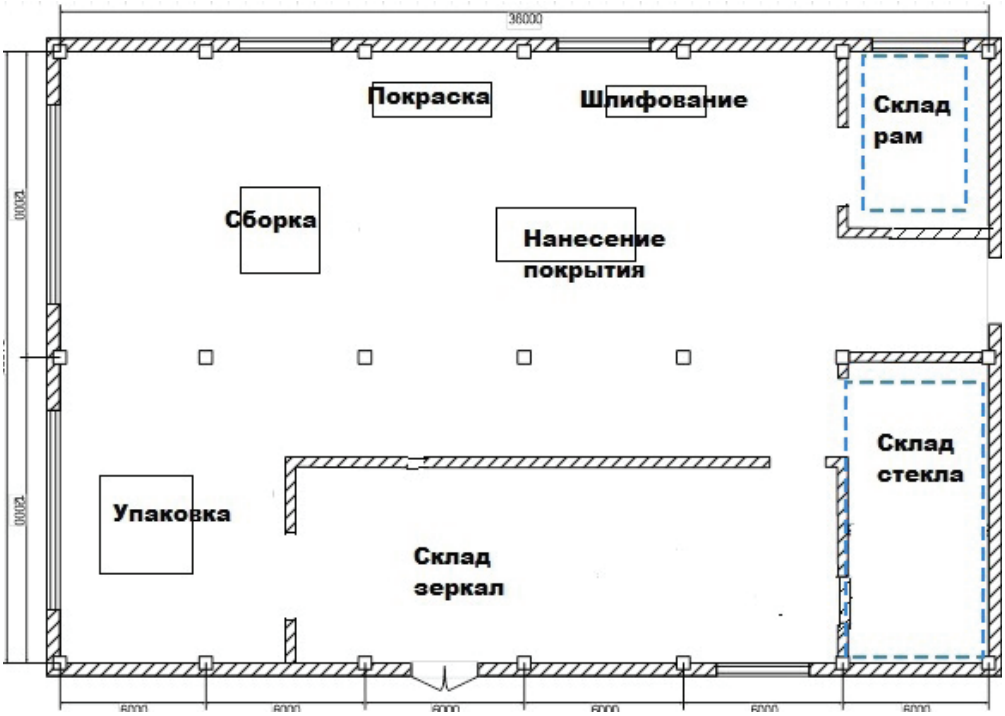

Рис. 3.13. Размещение элементов **Прямоугольный узел**

В свойствах элементов **Прямоугольный узел** задайте их имена: **node_frame** (для склада рам) и **node_glass** (для склада стекла).

Этап 4. Привязка мест хранения заготовок к производящим их элементам в модели

Привяжем элемент **Detal1** к складу рам. Для этого выделите элемент **Detal1** и задайте его свойства в разделе **Местоположение прибытия** (рис. 3.14).

Аналогичным образом привяжите элемент **Detal2** к складу стекла (рис. 3.15).

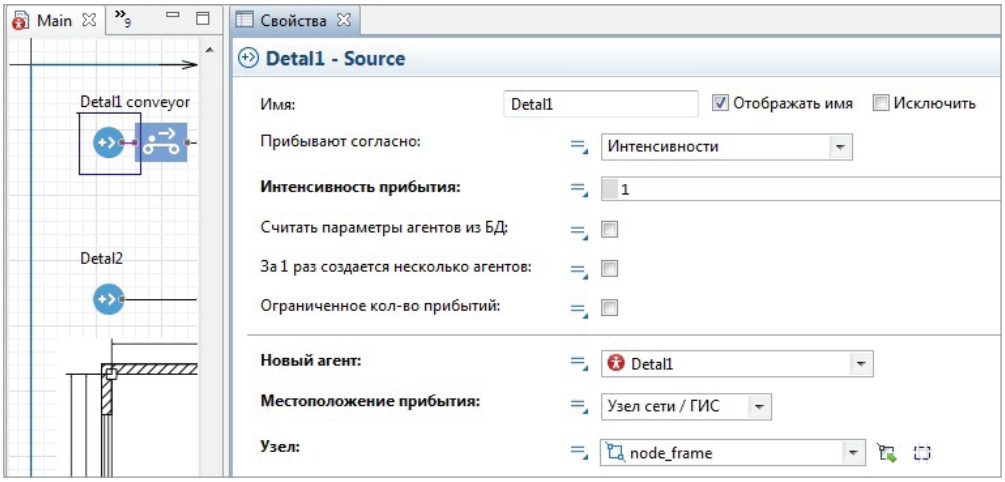

Рис. 3.14. Привязка склада рам к элементу **Detal1**

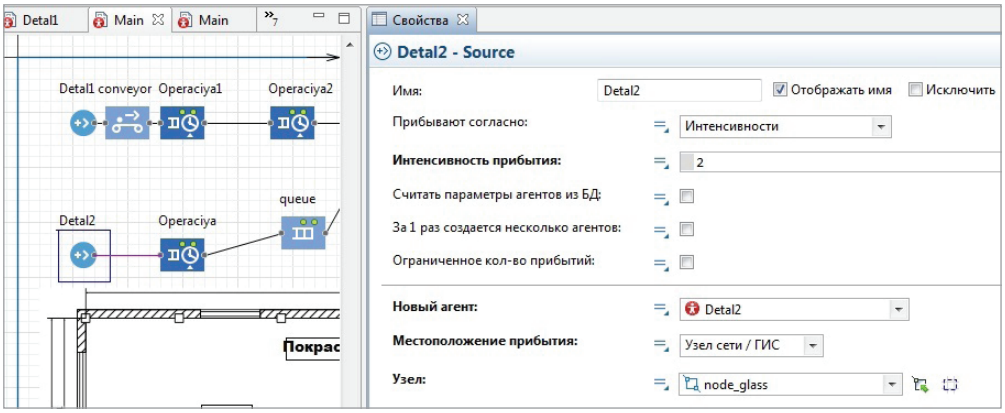

Рис. 3.15. Привязка склада стекол к элементу **Detal2**

Этап 5. Задание логики движения деталей до мест их обработки и сборки

Детали до мест их обработки и сборки доставляются конвейером. Для моделирования конвейера в модели используется элемент **Conveyor** библиотеки моделирования процессов (рис. 3.16).

Вставьте элементы **Conveyor** до элементов, моделирующих технологические операции в модели, как показано на рис. 3.17.

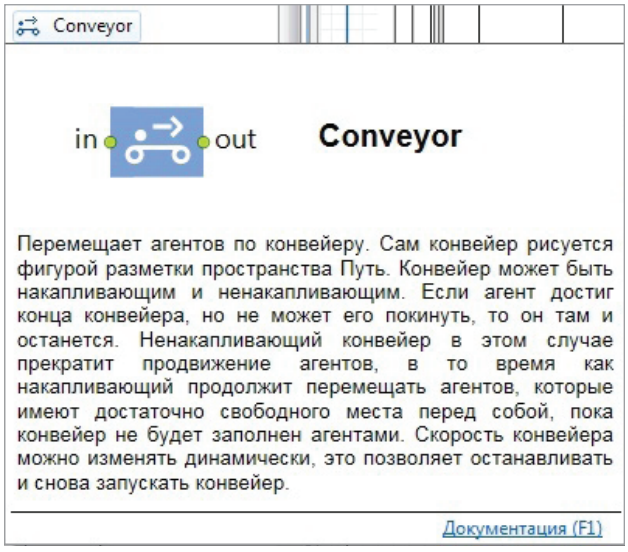

Рис. 3.16. Элемент **Conveyor**

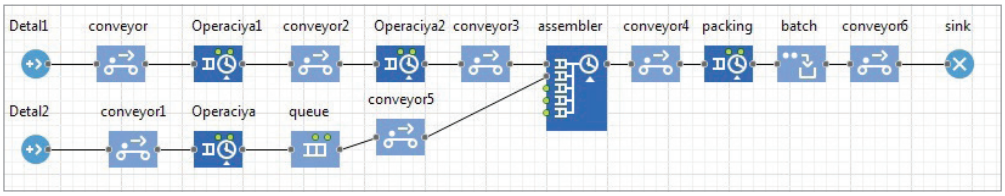

Рис. 3.17. Добавление конвейеров в модель

Этап 6. Задание путей движения деталей по цеху

Для задания пути движения заявки в модели используется элемент **Путь** библиотеки **Разметка пространства** (рис. 3.18). Для работы с элементом дважды щелкните на нем.

Задайте пути движения деталей и изделия, как показано на рис. 3.19.

В свойствах путей задайте имена (**path1** — со склада рам до первой операции, **patch2** — с первой операции до второй, **path3** — со второй операции на сборку, **patch4** — со склада стекла на операцию, **patch5** — с операции на сборку, **patch6** — со сборки на упаковку, **patch7** — с упаковки на склад зеркал), в разделе **Внешний вид** выберите тип **Конвейер** (рис. 3.20) и задайте его ширину 1 м.

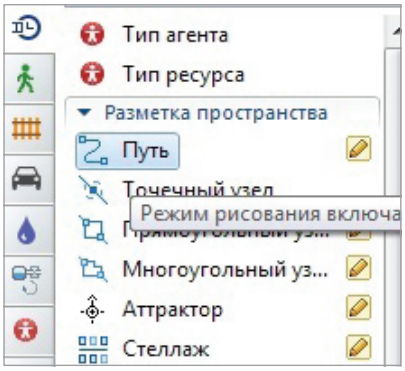

Рис. 3.18. Элемент **Путь**

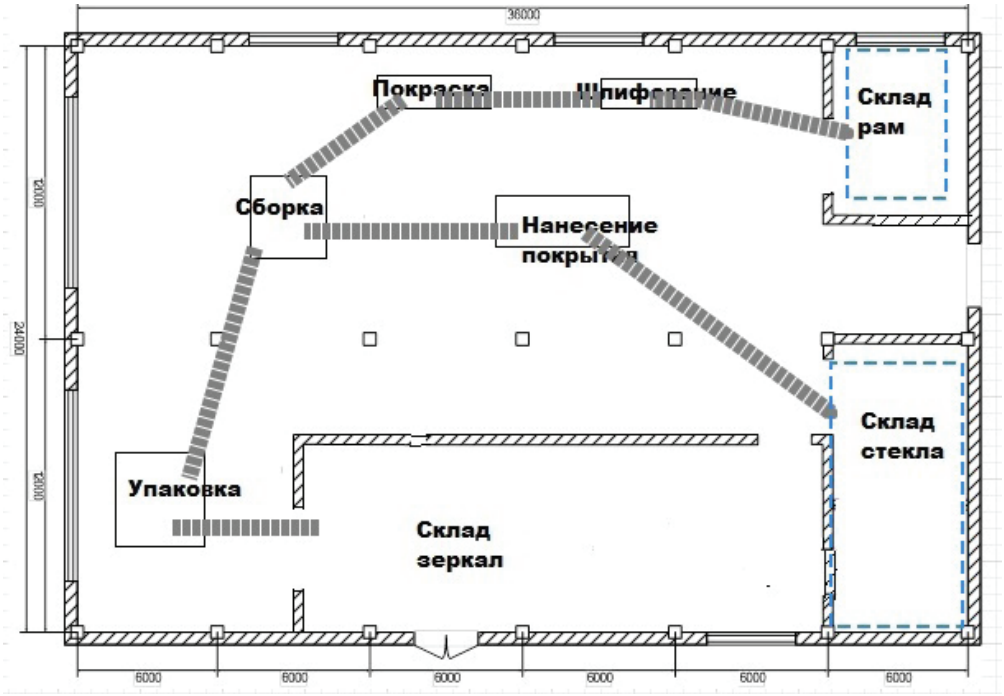

Рис. 3.19. Пути движения деталей и изделия

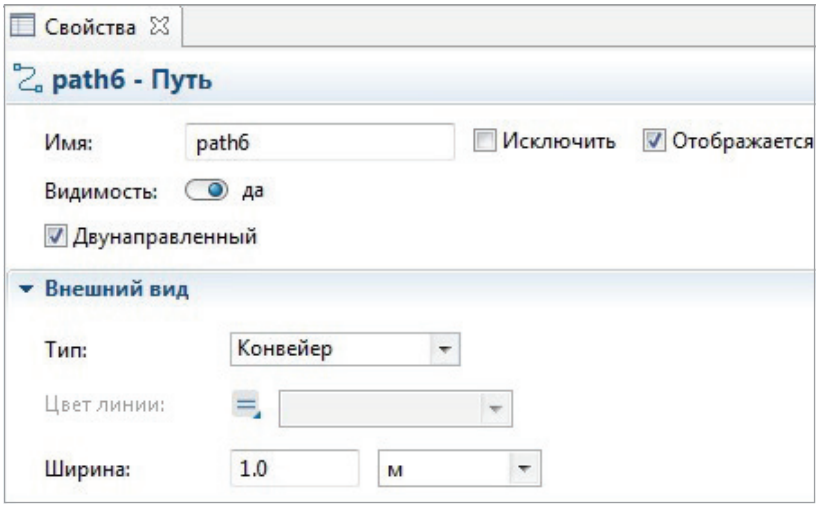

Рис. 3.20. Свойства пути

Этап 7. Привязка путей к конвейерам в логике модели

Выделите элемент **Conveyor**, стоящий между элементами **Detal1** и первой операцией, и задайте его свойства, как показано на рис. 3.21.

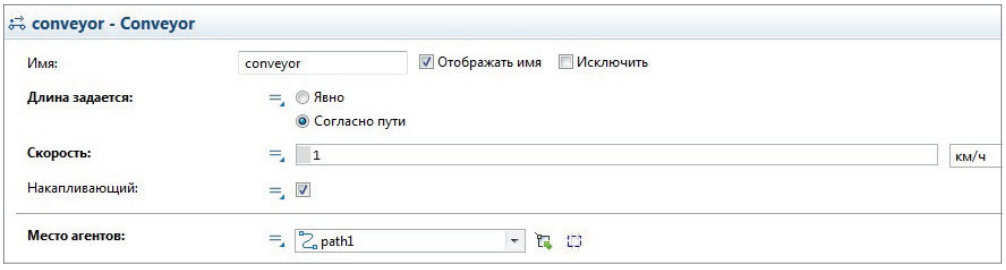

Рис. 3.21. Свойства первого конвейера

Аналогично задайте свойства всех остальных конвейеров.

Запустите модель. Если все пути были заданы верно, то по конвейерам должны двигаться рамы, стекло, зеркала и коробки.

Этап 8. Добавление мест для операции упаковки и склада готовых изделий

С помощью инструмента **Прямоугольный узел** задайте места для операции упаковки и склада готовых изделий, как показано на рис. 3.22.

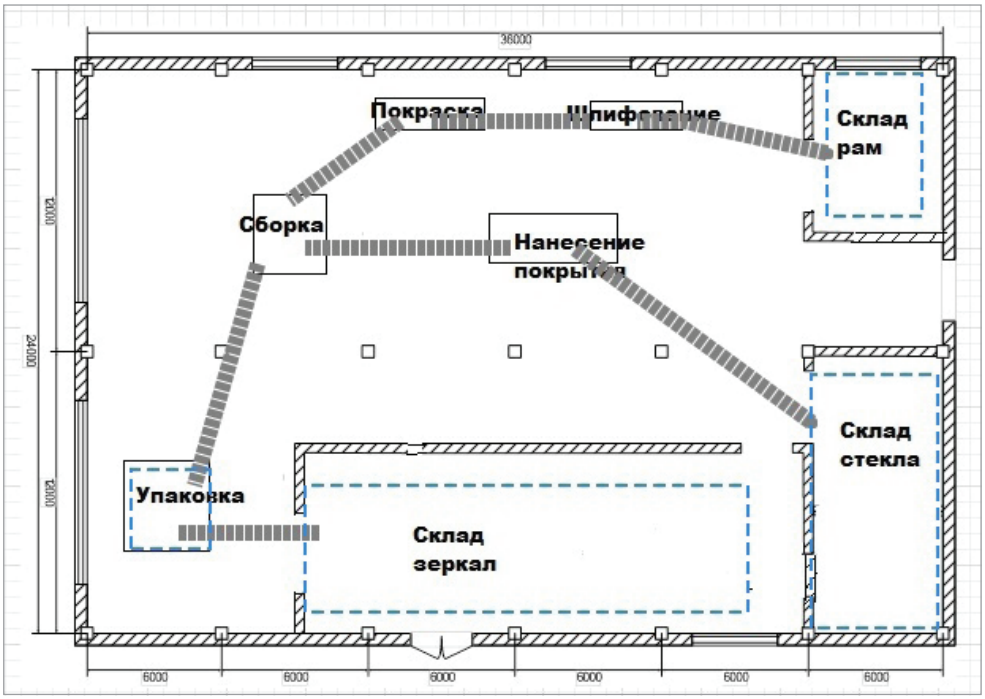

Рис. 3.22. Вставка мест для операции упаковки и склада готовых изделий

Задайте имена для этих областей (**node_mirrow** — для склада готовых изделий и **node_pack** — для операции упаковки).

Этап 9. Привязка операции упаковки к месту упаковки

Выделите элемент **Packing** и задайте в его свойствах **Место агентов**, как показано на рис. 3.23.

Теперь нужно привязать к месту еще и элемент **bath**. Зайдите в его свойства и задайте в них место агентов и место новой партии, как показано на рис. 3.24.

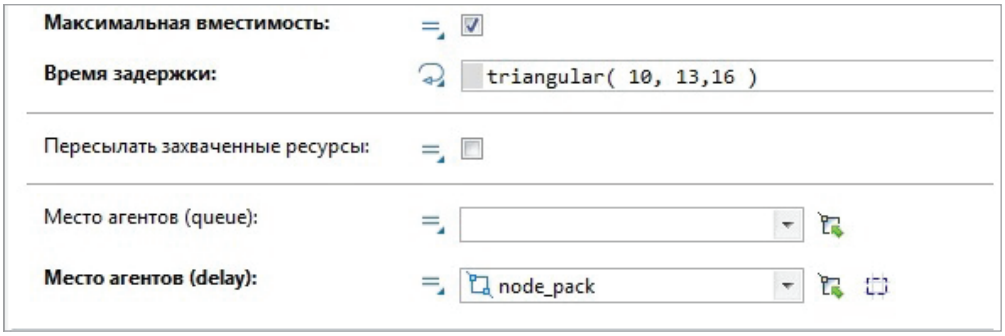

Рис. 3.23. Свойства агента **Packing**

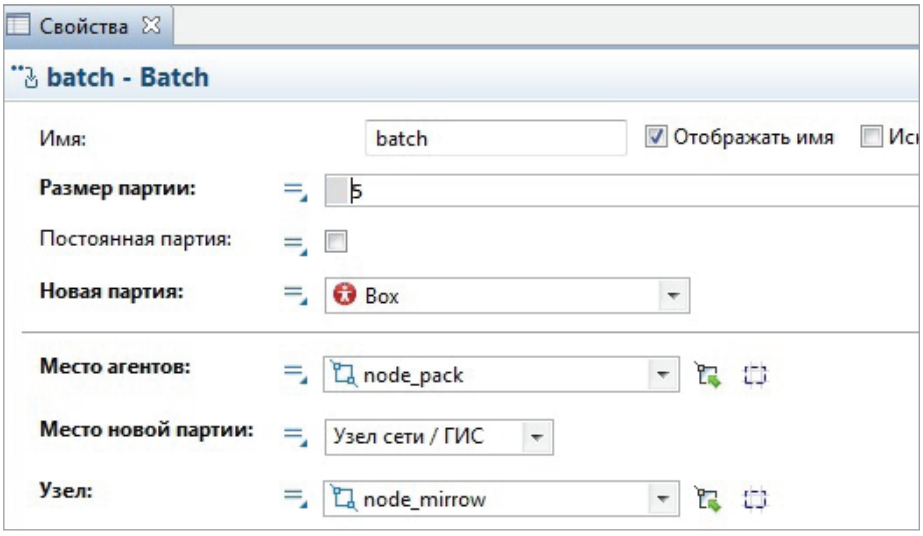

Рис. 3.24. Задание места для агентов элемента **batch**

Этап 10. Имитация задержки коробок с зеркалами на складе

Поскольку коробки с зеркалами не сразу увозятся из цеха, то для моделирования их задержки на складе используется элемент **Delay** библиотеки моделирования процессов (рис. 3.25).

Вставьте элемент **Delay** перед элементом **sink** (рис. 3.26).

Задайте свойства элемента **Delay**, как показано на рис. 3.27.

Запустите модель. Должна получиться анимация, как показано на рис. 3.28.

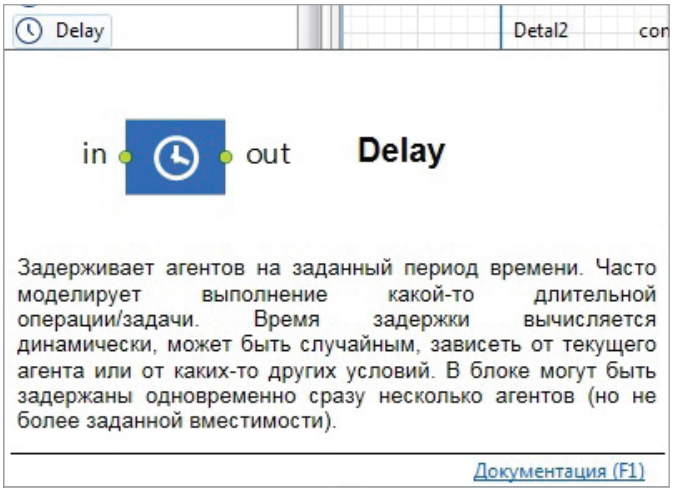

Рис. 3.25. Элемент **Delay**

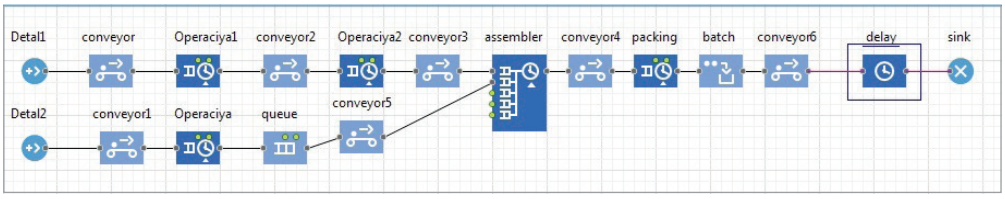

Рис. 3.26. Конечная модель логики процесса сборки зеркал

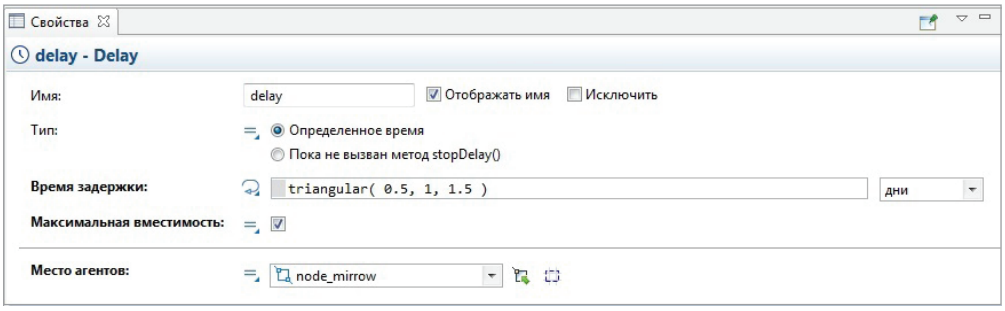

Рис. 3.27. Свойства элемента **Delay**

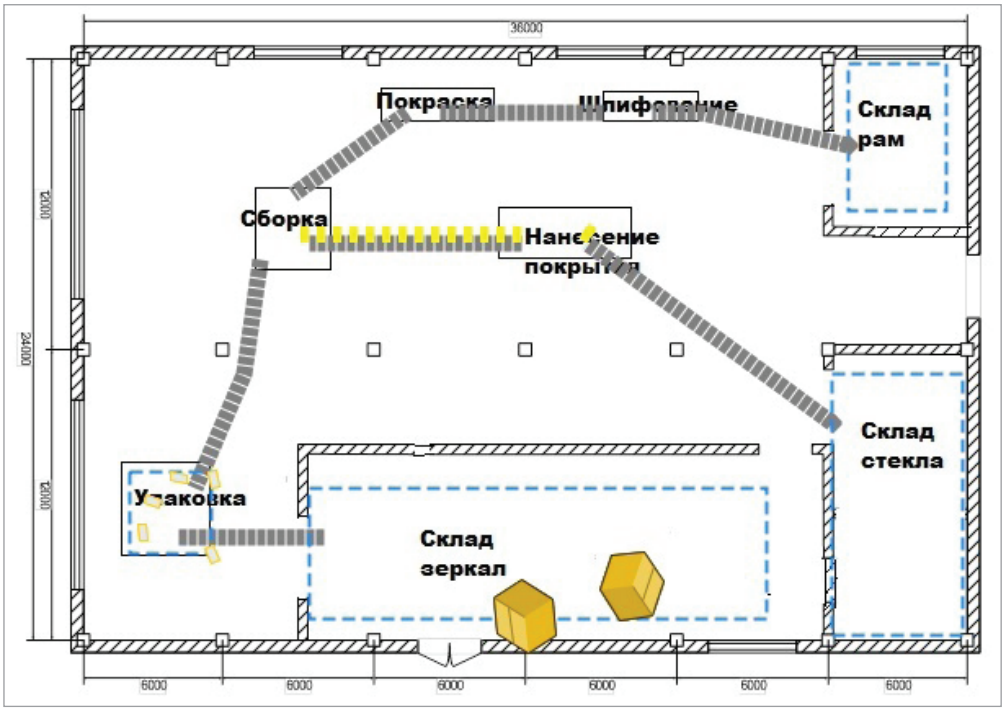

Рис. 3.28. Анимация модели

Лабораторная работа № 4

Сбор статистики в AnyLogic 8.1

Задача

Дополним разработанную ранее модель сборочного цеха статистическими данными, а именно соберем статистику по занятости рабочих и роботов на операциях в процессе сборки. Инструменты для работы со статистикой собраны в библиотеке **Статистика** вкладки **Палитра** (рис. 4.1).

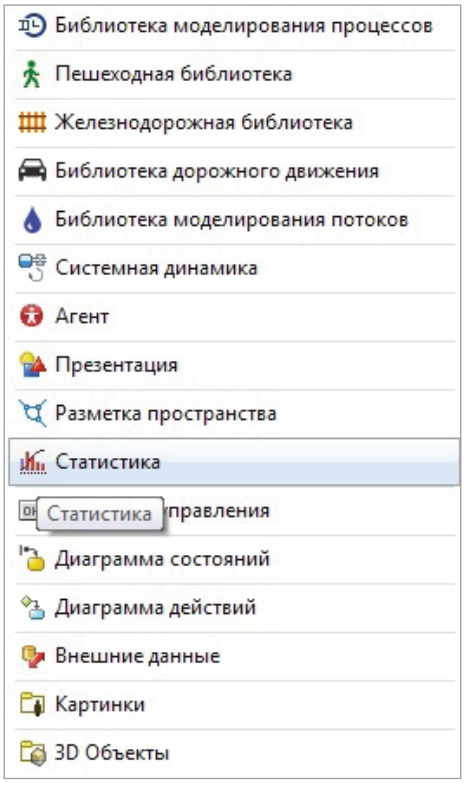

Рис. 4.1. Библиотека **Статистика**

Решение

Этап 1. Задание инструментов для накопления данных по занятости роботов и рабочих

Для накопления статистических данных в ходе выполнения модели используется элемент **Набор данных** (рис. 4.2).

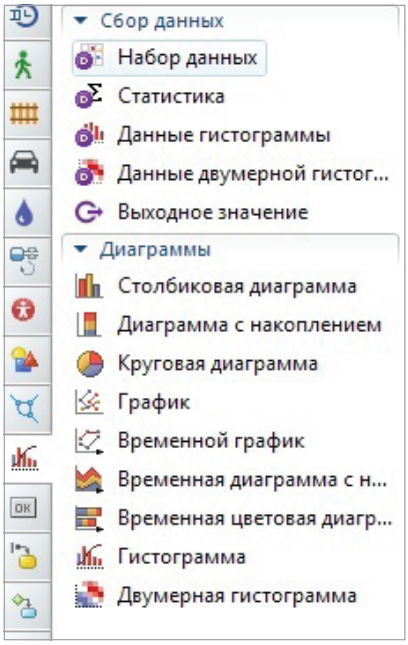

Рис. 4.2. Элемент **Набор данных**

Перетащите его на рабочее поле модели и задайте его свойства так, как показано на рис. 4.3.

Здесь задается имя набора данных и вызывается функция **utilization ()**, которая показывает занятость ресурса **Worker1**. Полученные из функции значения сохраняются как двухмерный массив данных, в котором записывается для каждого момента времени моделирования значение функции **utilization ()**.

Повторите операцию для другого набора данных по сбору информации по занятости рабочего на операции по обработке второй детали (рис. 4.4).

Аналогичным образом задайте наборы данных для сбора информации по занятости роботов (рис. 4.5–4.6).

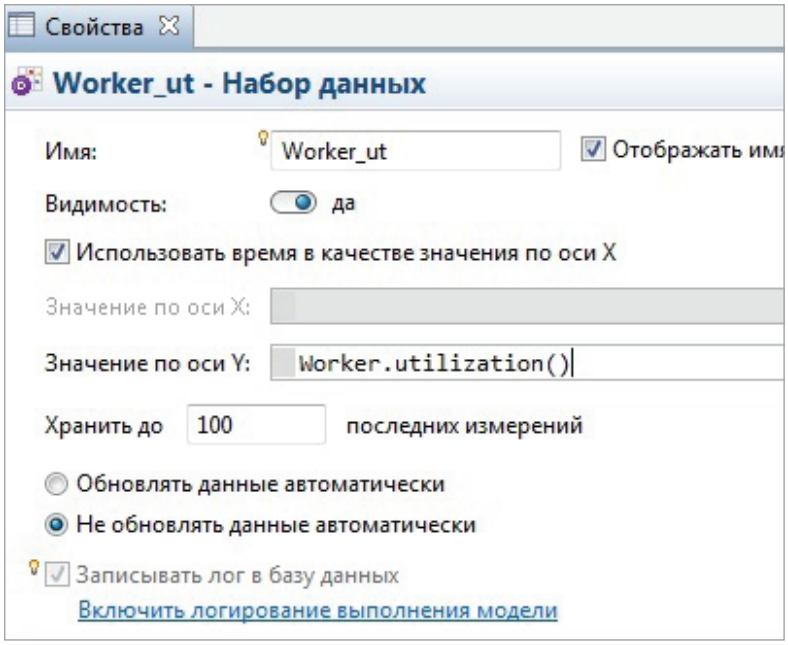

Рис. 4.3. Свойства набора данных для сбора статистики по занятости рабочего на второй технологической операции по обработке первой детали

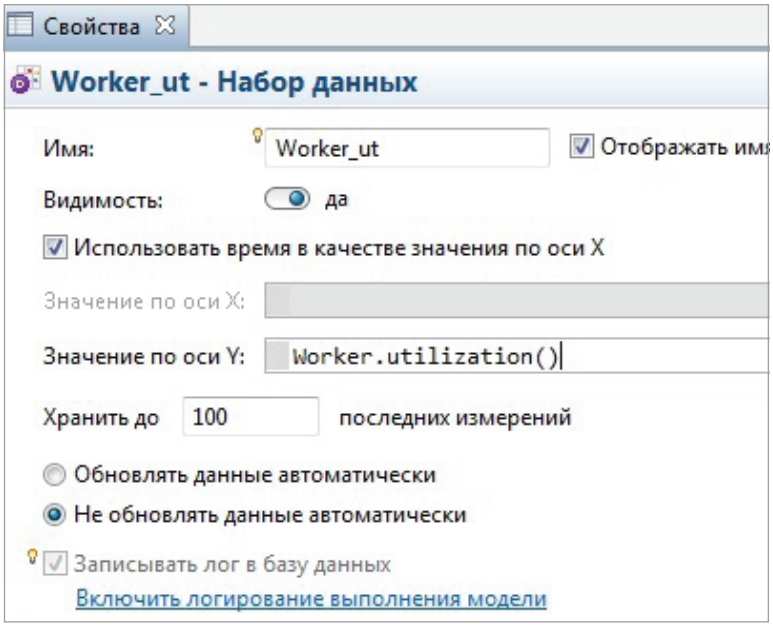

Рис. 4.4. Задание набора данных для рабочего, занятого на операции по обработке второй детали

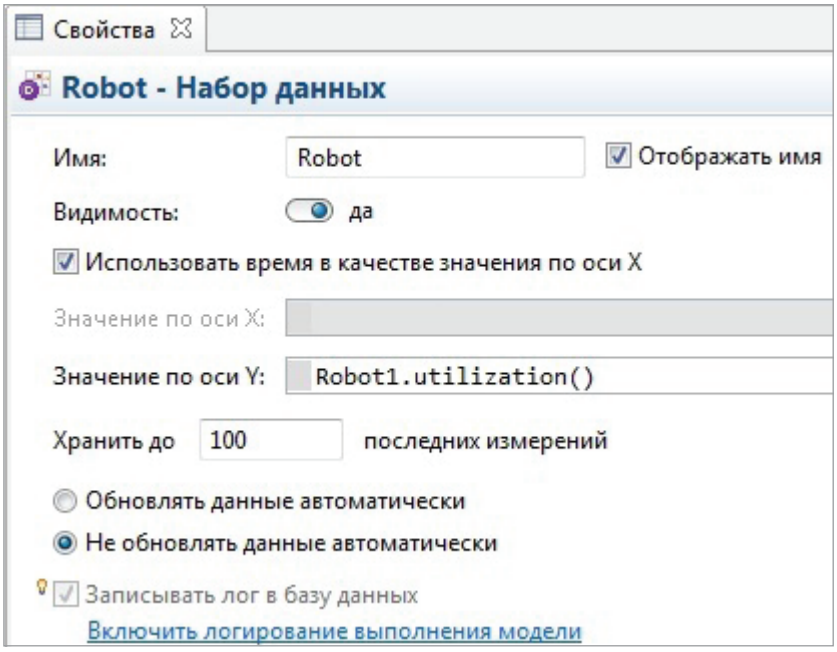

Рис. 4.5. Набор данных для сборки информации по занятости робота на первой операции по обработке первой детали

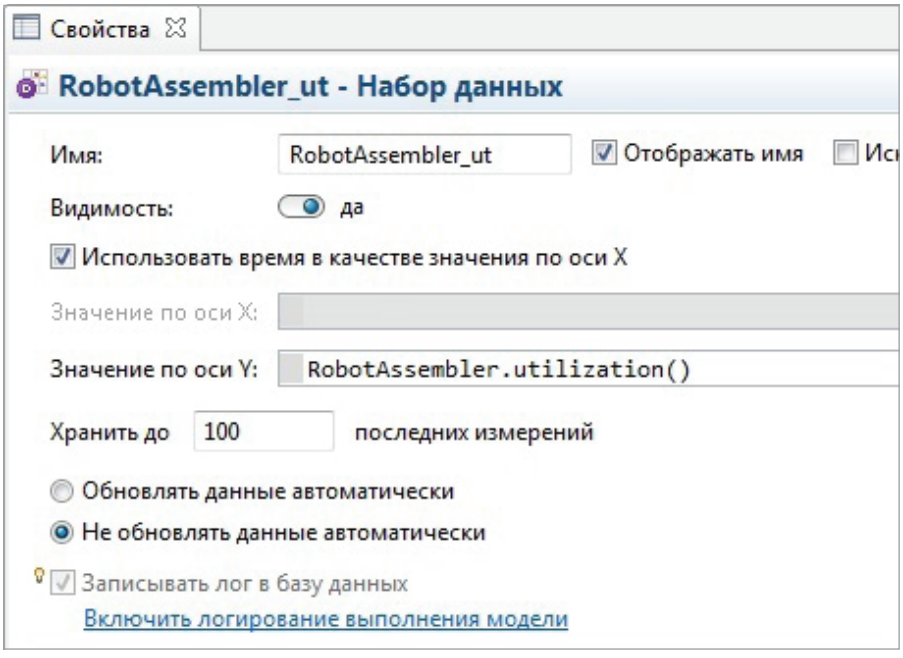

Рис. 4.6. Набор данных для сбора информации по занятости робота-сборщика

Этап 2. Задание графического отображения накапливаемых в ходе работы модели статистических данных

Инструменты для графического отображения статистических данных собраны во вкладке **Диаграммы** библиотеки **Статистика** (рис. 4.7).

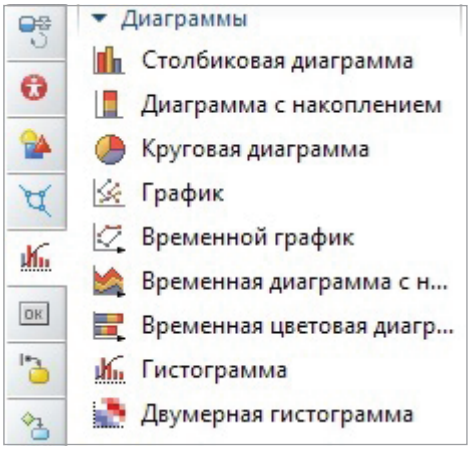

Рис. 4.7. Раздел **Диаграммы**

Отобразим занятость рабочих с помощью **Временного графика**. Этот элемент по оси X всегда откладывает модельное время, а по оси Y то, что задаст пользователь.

Перетащите элемент **Временной график** на рабочее поле модели и задайте его свойства так, как показано на рис. 4.8. Для добавления нового набора данных в график нажмите +.

Для отображения занятости роботов используем **Круговую диаграмму**. Перетащите элемент **Круговая диаграмма** и задайте ее свойства так, как показано на рис. 4.9

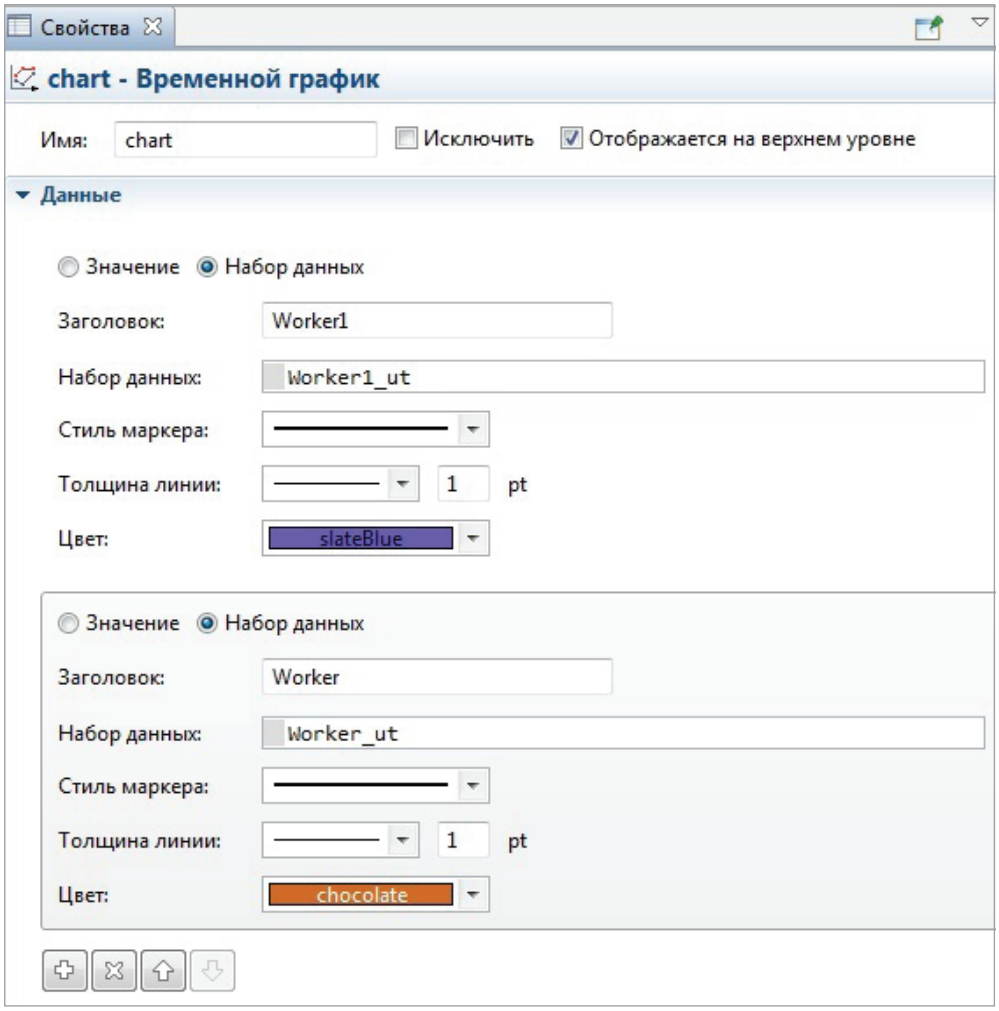

Рис. 4.8. Свойства элемента **Временной график**

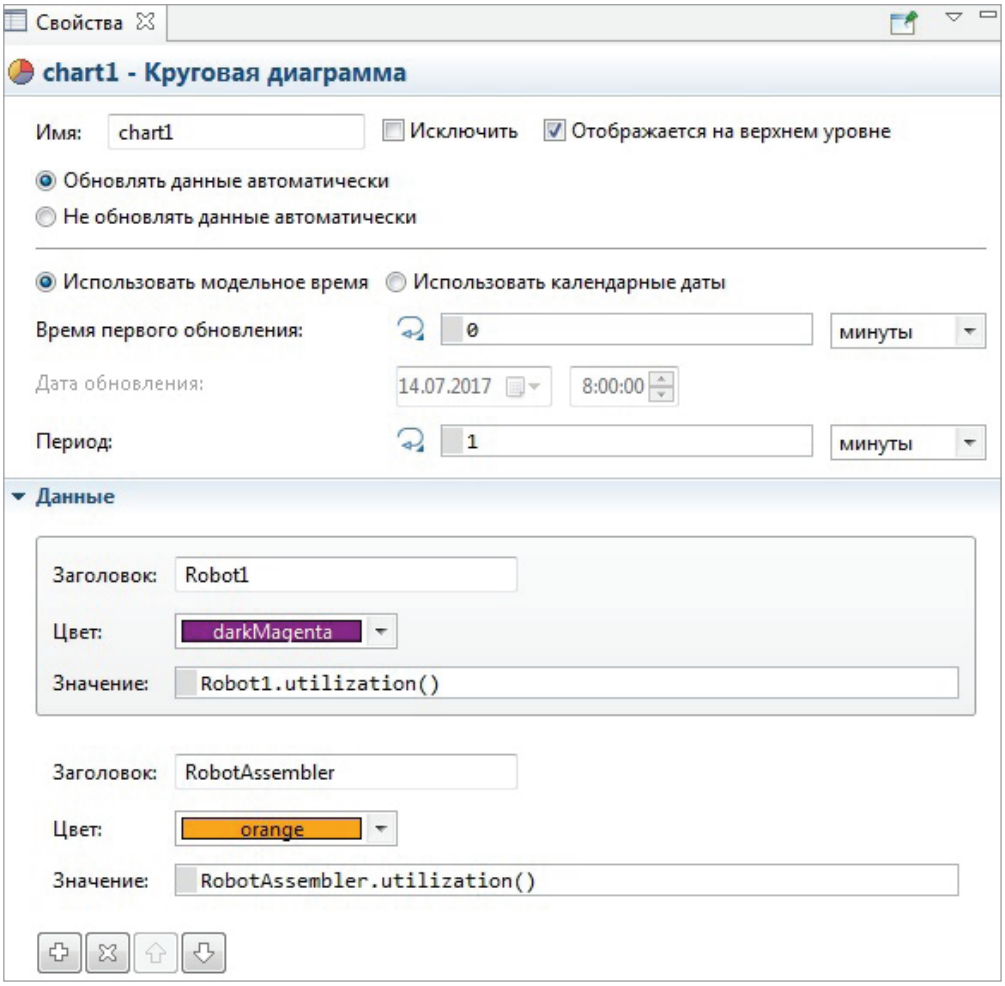

Рис. 4.9. Свойства элемента **Круговая диаграмма**

Этап 3. Добавление часов в модель

Очень удобно наблюдать за модельным временем на часах, расположенных в самой модели.

Элемент **Часы** находится в библиотеке **Картинки** (рис. 4.10).

Перетащите элемент на рабочее поле модели. Запустите модель (рис. 4.11).

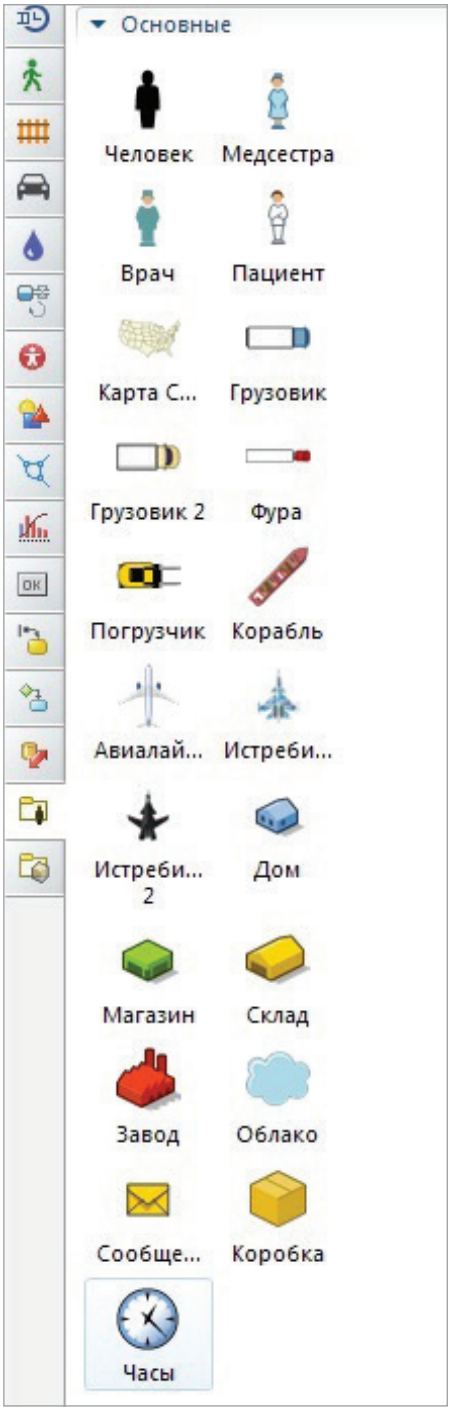

Рис. 4.10. Библиотека **Картинки**

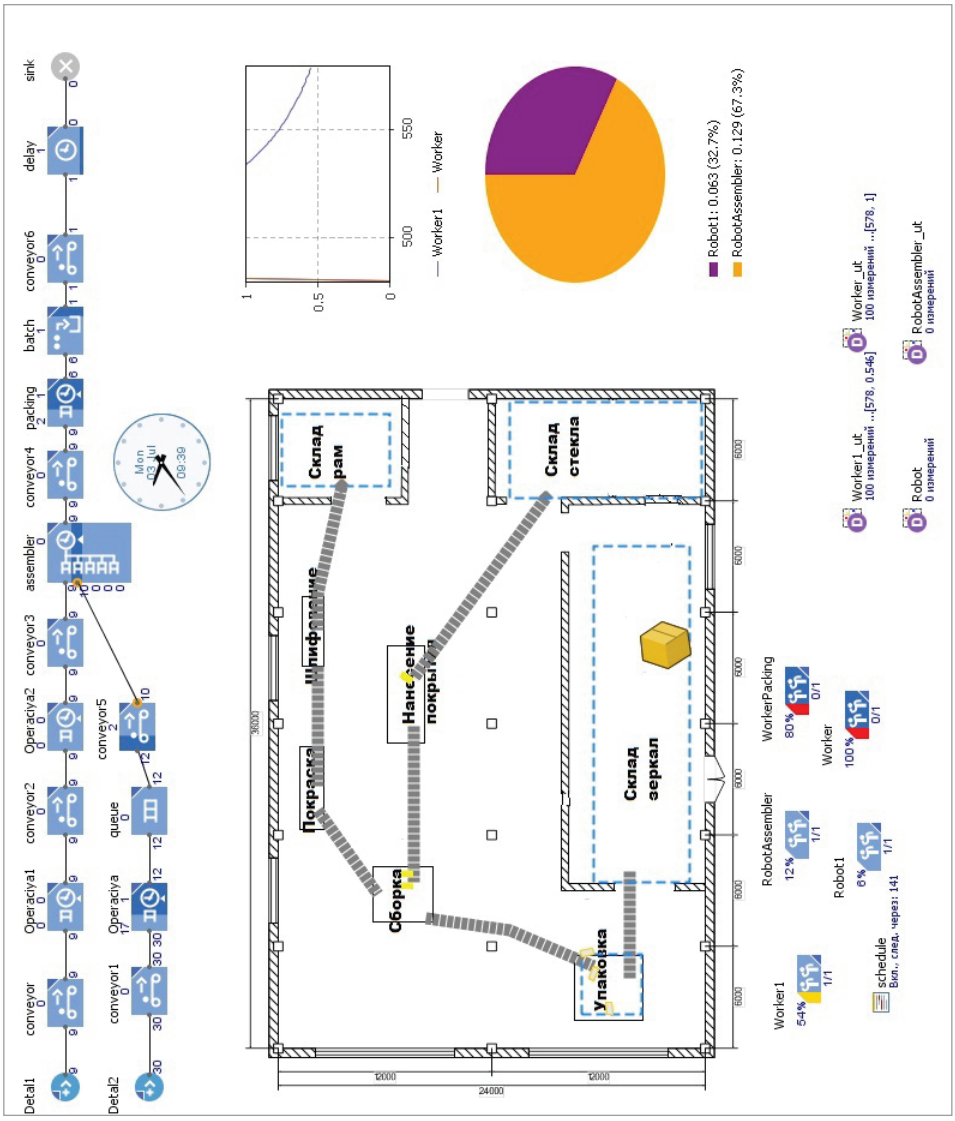

Рис. 4.11. Результат работы модели

Лабораторная работа № 5

Соединение нескольких моделей в одну

Задача

Вернемся к первой лабораторной работе, в которой из двух деталей, находящихся на складе, производилась сборка одного изделия. Промоделируем отдельно склад деталей, сборку и склад готовых изделий. Затем соединим эти модели в одном проекте.

Решение

Этап 1. Создание агентов модели

Создайте новую модель в среде AnyLogic и назовите ее **ModernSborka**. В качестве единиц модельного времени задайте минуты (рис. 5.1).

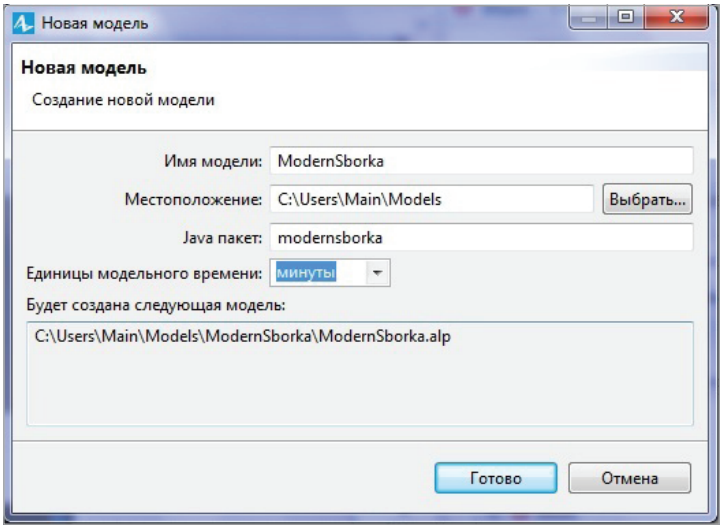

Рис. 5.1. Создание новой модели

В модели создайте нового единственного агента (рис. 5.2), дайте ему имя **Storage** и задайте анимацию (склад).

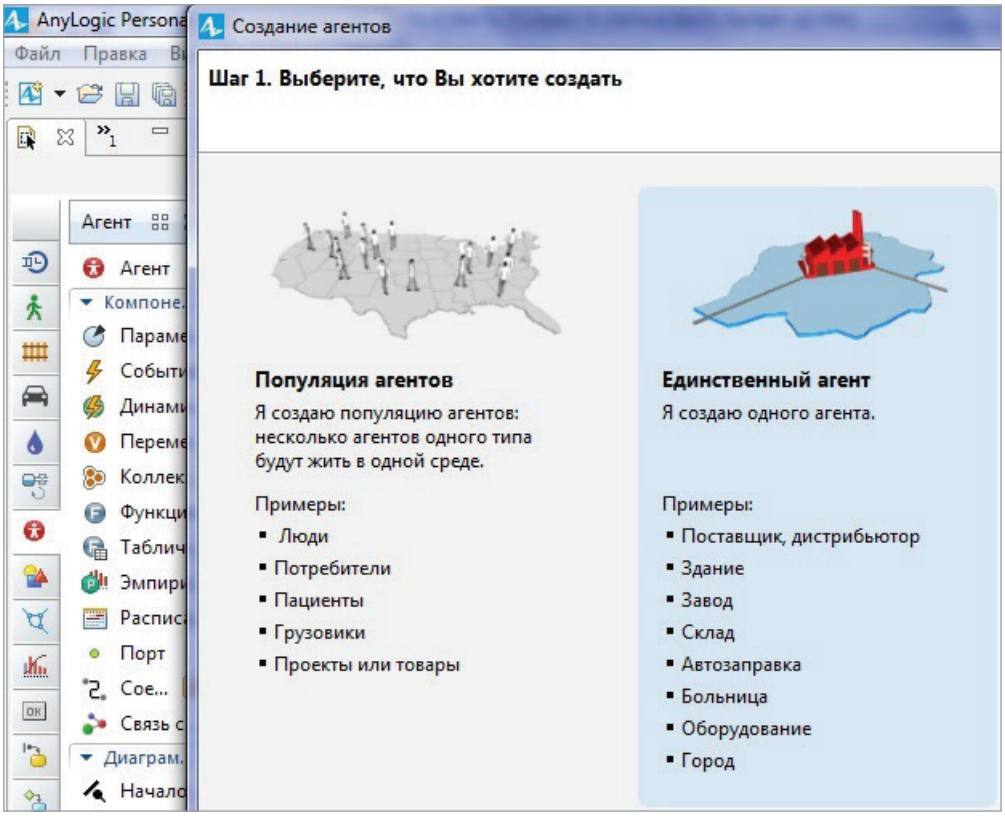

Рис. 5.2. Создание агента **Storage**

Повторите операцию и создайте единственных агентов **Plant** (анимация **Завод**), **Storage_Production** (анимация **склад1**), **Detal1** (анимация **сфера**), **Detal2** (анимация **конус**), **Izdelie** (анимация **пирамида**) и **Box** (анимация **Коробка закрытая**). Задайте координаты для агентов. Должно получиться так же, как на рис. 5.3.

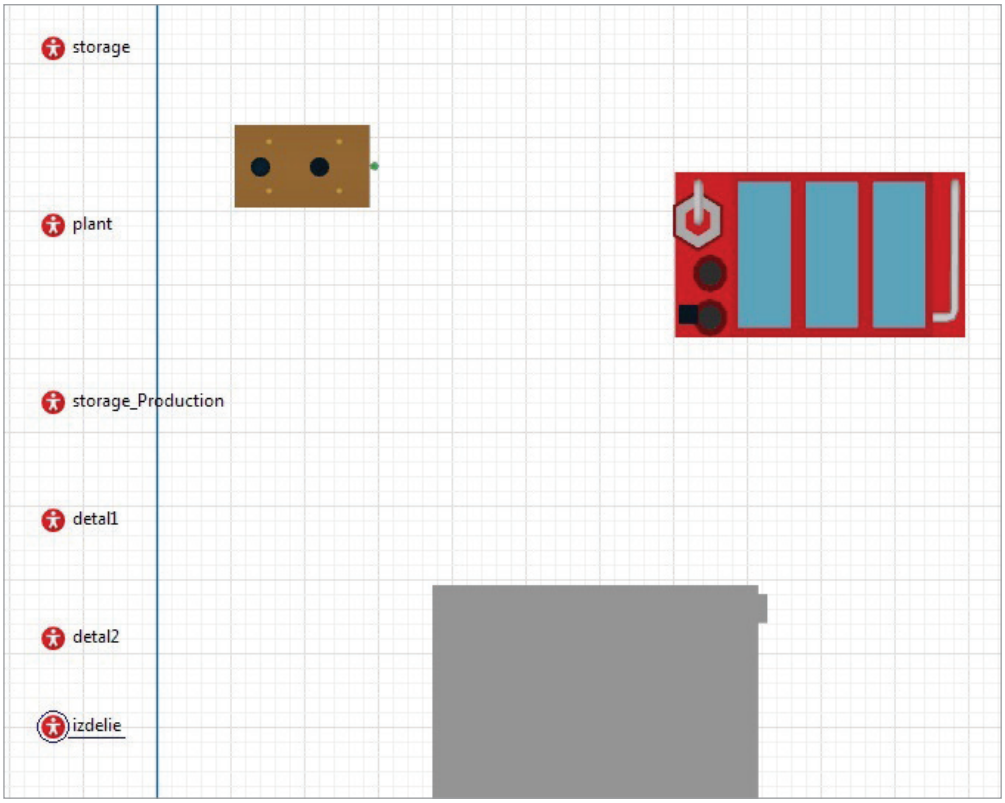

Рис. 5.3. Размещение агентов в модели

Этап 2. Моделирование агента Storage

Зайдите в агент **Storage**. Перейдите в библиотеку моделирования процессов, перетащите элемент **Source** на рабочее поле агента **Storage** и задайте его свойства так, как показано на рис. 5.4.

Повторите операцию для моделирования поставок деталей второго типа. Должно получиться так, как на рис. 5.5.

Для перемещения агентов в модели можно использовать кроме элемента **Conveyor**, который был использован для перемещения деталей в лабораторной работе № 1, элемент **MoveTo** (рис. 5.6).

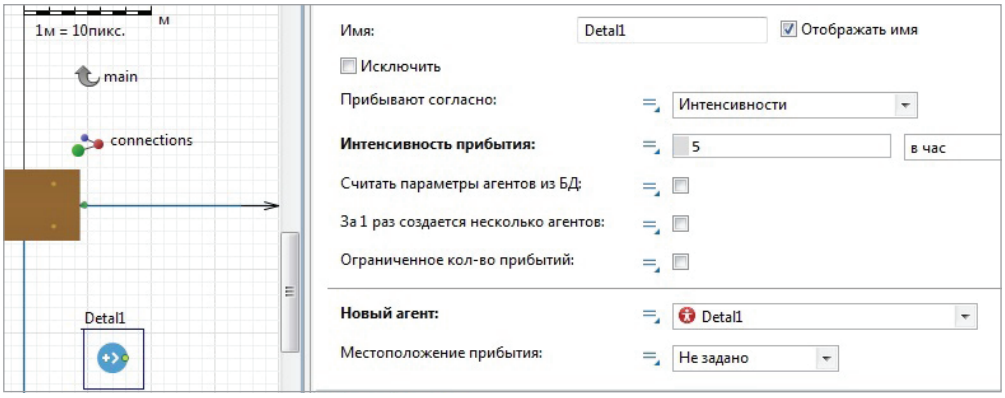

Рис. 5.4. Моделирование поставок деталей **Detal1**

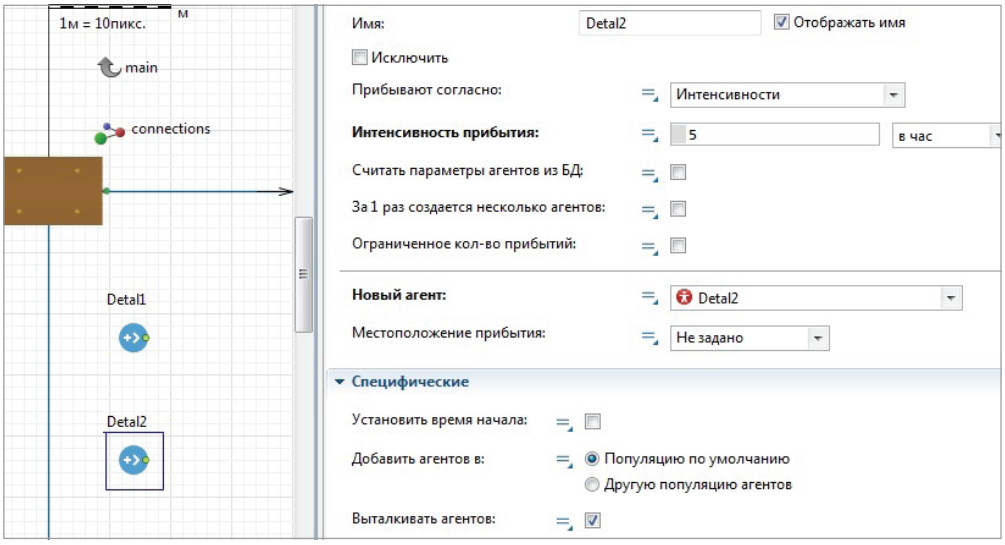

Рис. 5.5. Моделирование поставок деталей второго типа

В свойствах этого элемента указывается место назначения. Местом назначения может быть как узел в модели, так и другой агент. В этой работе детали будут отправляться к другому агенту (агенту **Storage**).

Перетащите элемент **MoveTo** к элементу **Detal1** и задайте его свойства так, как показано на рис. 5.7.

Здесь элемент **MoveTo** отправляет агент **Detal1** к агенту **Plant**.

Повторите эту операцию для элемента **Detal2** (рис. 5.8).

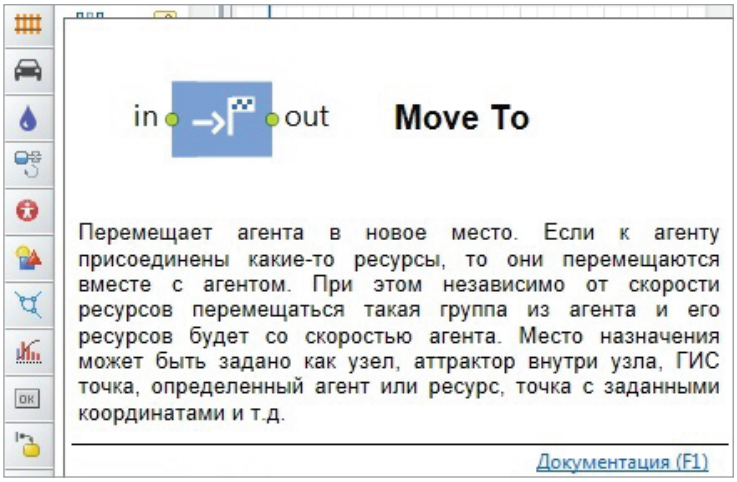

Рис. 5.6. Элемент **MoveTo**

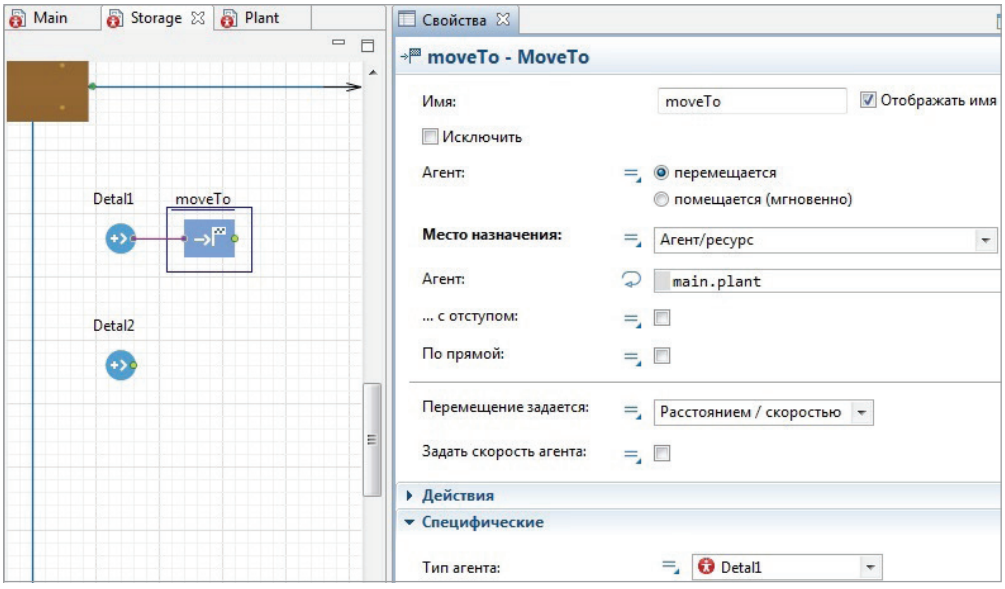

Рис. 5.7. Привязка и свойства элемента **MoveTo**

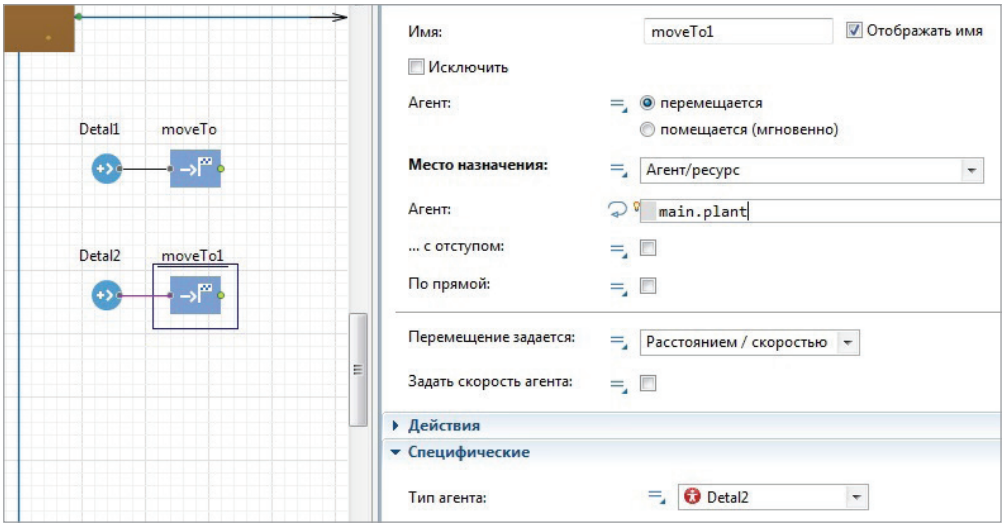

Рис. 5.8. Привязка и свойства элемента **MoveTo1**

Теперь агенты **Detal** должны покинуть агент **Storage** и попасть в агент **Plant**. Использование только элемента **MoveTo** для этого не достаточно, нужно создать переход из агента **Storage** в агент **Plant**. Для этого сначала создается выход из агента **Storage**.

Создание выхода из агента Storage

Перетащите значок **Склад** из библиотеки **Картинки** на рабочее поле агента **Storage** (рис. 5.9).

Перетащите 2 элемента **Порт** из агентной библиотеки на рабочее поле агента **Storage** и назовите один **port_Detal1**, а другой — **port_Detal2**. Соедините порты с элементами **MoveTo** (рис. 5.10).

Значок **Склад** и порты автоматически появятся в агенте **main**.

Этап 3. Моделирование агента Plant

Создание входа в агент из агента Storage

Перейдите в агент **Plant** и перетащите значок **Завод** из библиотеки **Картинки** (рис. 5.11).

Перетащите 2 элемента **Порт** из агентной библиотеки и назовите один **port_Detal1**, второй — **port_Detal2**, как показано на рис. 5.12.

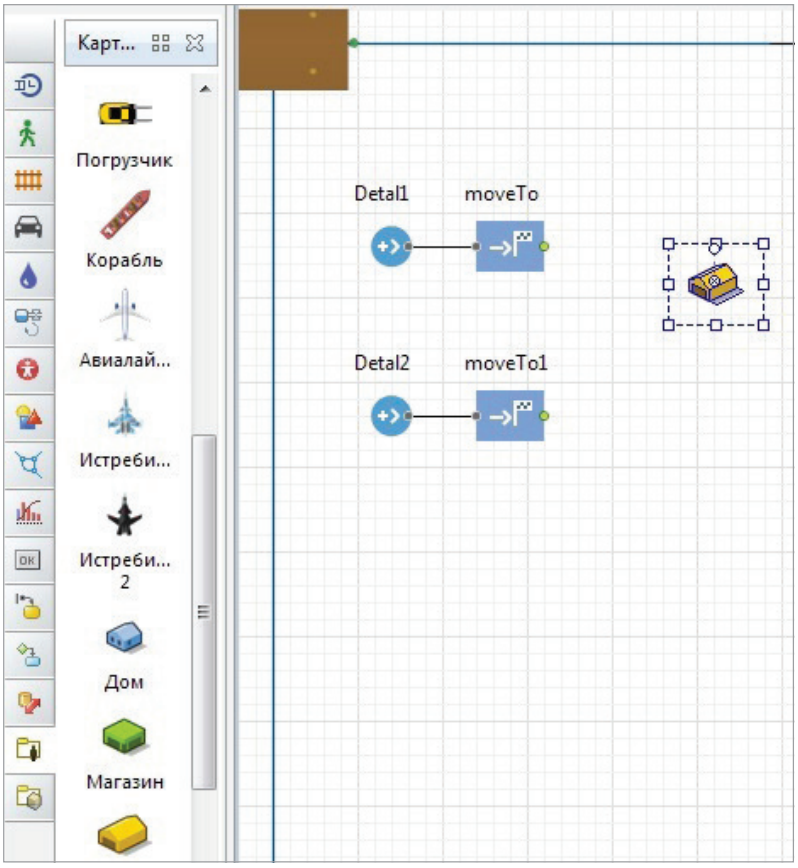

Рис. 5.9. Размещение агента **Storage**

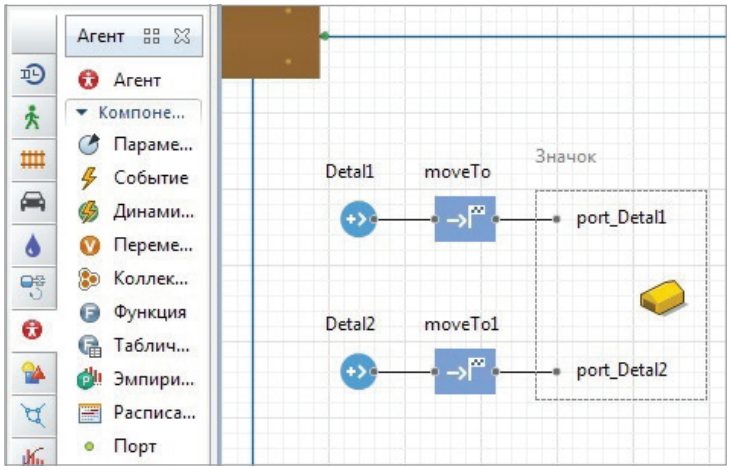

Рис. 5.10. Присоединение портов к агенту **Storage**

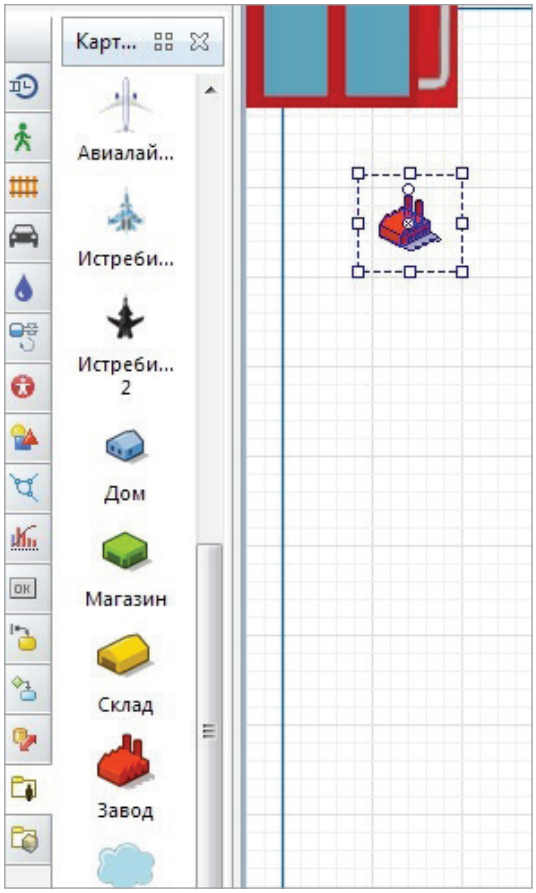

Рис. 5.11. Создание входа в агент **Storage**

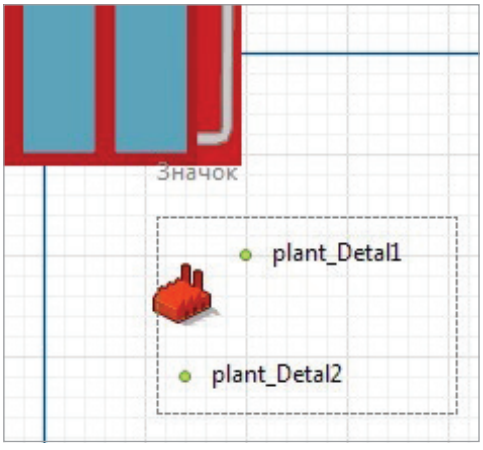

Рис. 5.12. Размещение портов в агенте **Storage**

Теперь нужно соединить входы в агент **Plant** с выходами из агента **Storage**. Делается это в агенте **main**.

Перейдите в агент **main** и соедините порты с помощью элемента **Соединитель** (он выбирается двойным щелчком мыши), как показано на рис. 5.13.

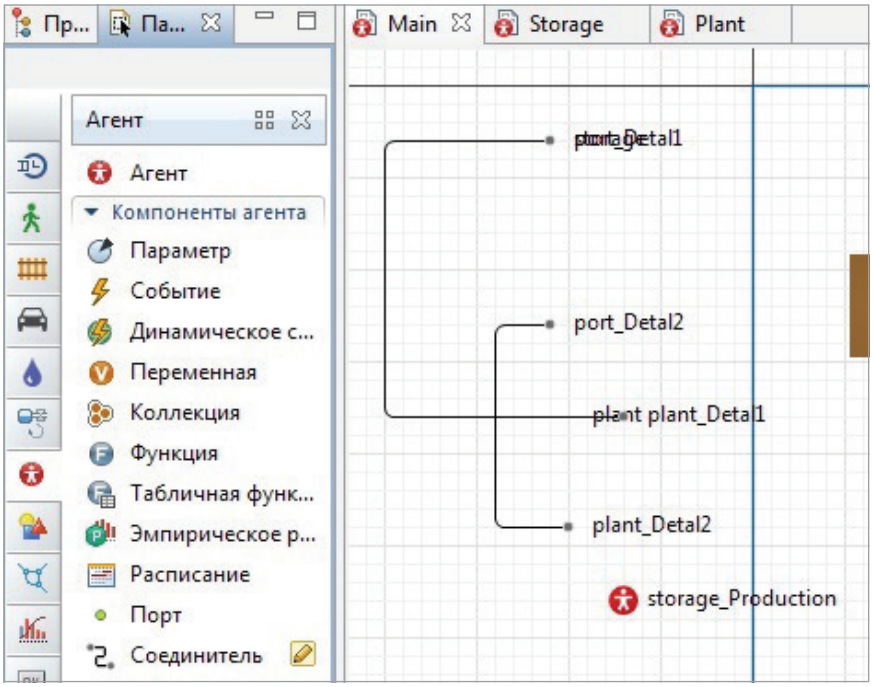

Рис. 5.13. Соединение портов

Моделирование процесса сборки изделия

Теперь из поступивших в агент **Plant** деталей собирается изделие. Причем первая деталь подвергается двум операциям, вторая — одной. Первая технологическая операция над первой деталью длится от 3 до 5 минут и выполняется 1 роботом. Вторая технологическая операция с первой деталью длится от 4 до 8 минут и выполняется 1 рабочим, который работает согласно расписанию (с 8 до 17 по рабочим дням с перерывом на обед с 12 до 13). Технологическая операция по обработке второй детали длится от 6 до 10 минут и выполняется рабочим. Сборка изделия выполняется роботом и длится от 6 до 12 минут. Изделия после сборки упаковываются по 5 штук. Упаковка изделий осуществляется рабочим и длится от 10 до 16 минут.

Перейдите в агент **Plant** и соедините порт **plant_Detal1** с элементом **Service**, который моделирует первую технологическую операцию, а порт **plant_Detal2** — с элементом, моделирующим операцию по обработке второй детали. Задайте их свойства, как показано на рис. 5.14 и 5.15 (обратите внимание на то, что в операции должны поступать соответствующие агенты).

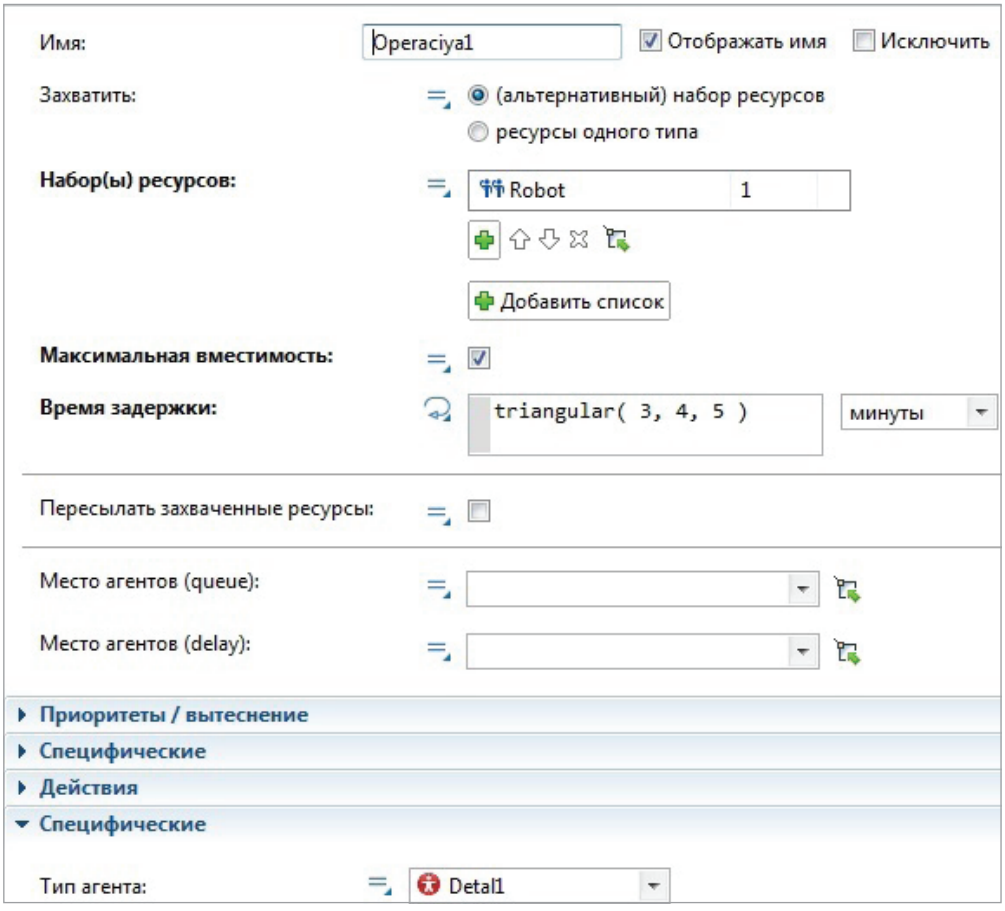

Рис. 5.14. Свойства первой технологической операции над первой деталью

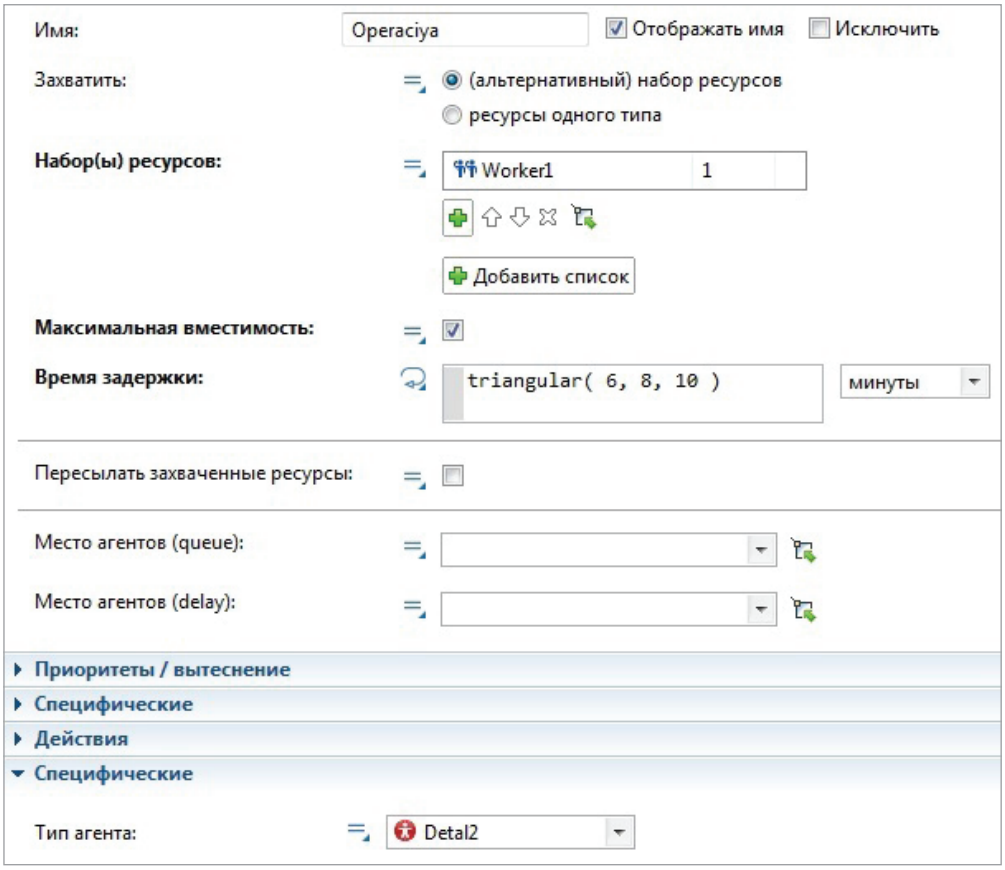

Рис. 5.15. Свойства операции над второй деталью

Поскольку разработка этой модели подробно приведена в лабораторной работе № 1, на рис. 5.16 приведен только конечный результат.

Обратите внимание, что на вход в операцию **Assembler** подается 2 агента — **Detal1** и **Detal2**, а на выходе получается агент **Izdelie** (рис. 5.17).

Также обратите внимание на то, что на вход в блок **batch** приходит агент **Izdelie**, а выходит агент **Box** (рис. 5.18).

Также убедитесь, что на вход в элемент **MoveTo** приходит агент **Box**.

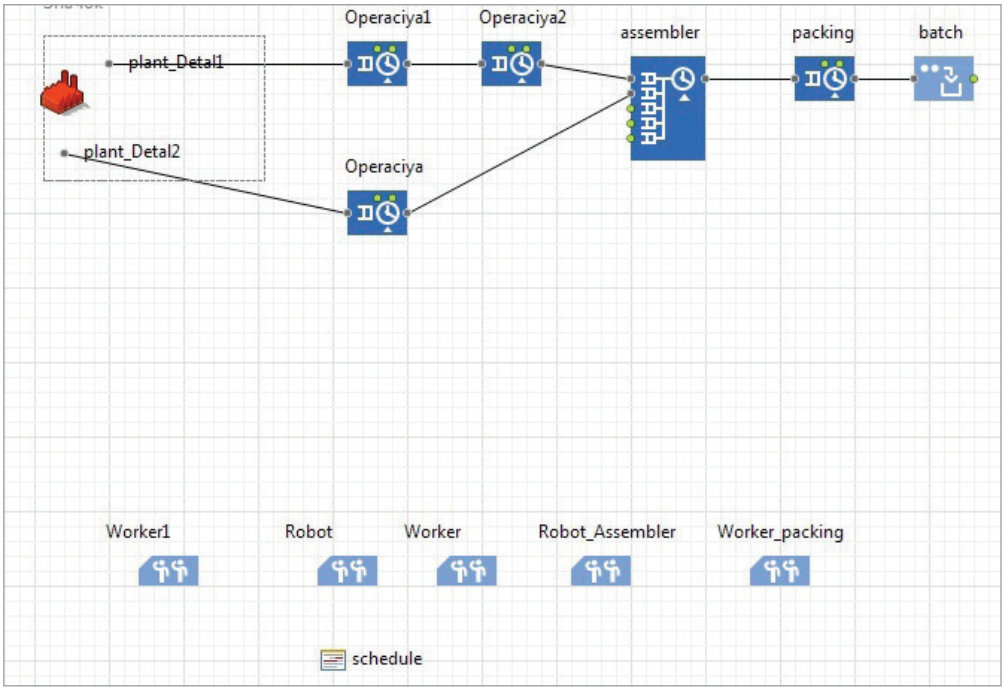

Рис. 5.16. Модель сборки

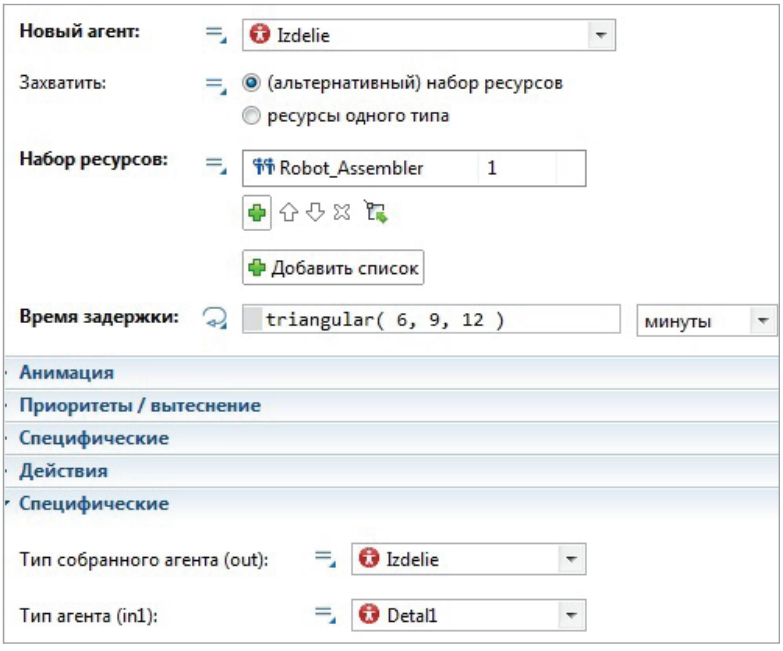

Рис. 5.17. Свойства операции **Assembler**

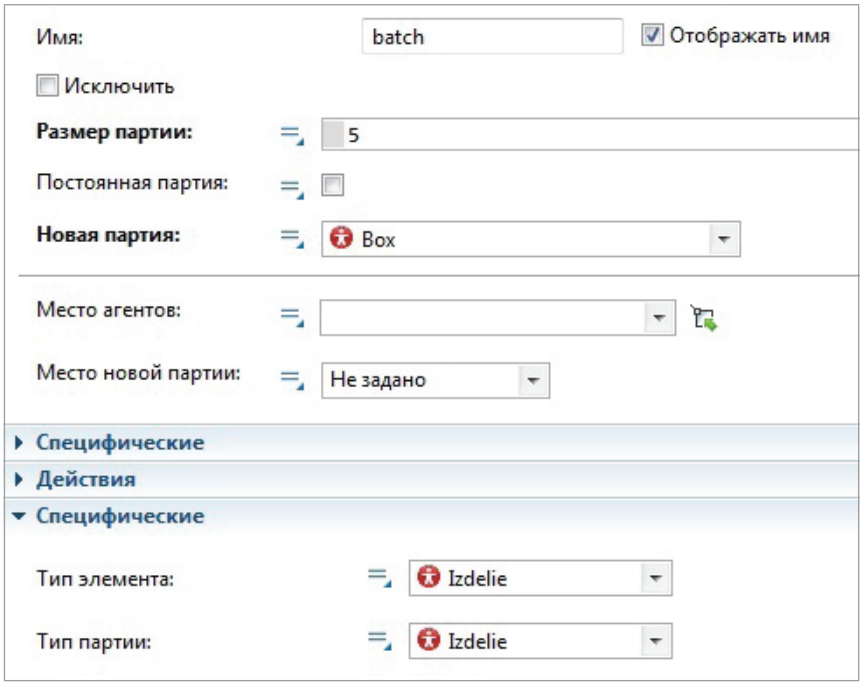

Рис. 5.18. Свойства блока **batch**

Моделирование выхода из агента Plant в агент Storage_Production

Выход из агента промоделируем с помощью элемента **MoveTo** и порта. Перетащите элемент **MoveTo** на рабочее поле агента и соедините его с элементом **batch**. Задайте его свойства, как показано на рис. 5.19.

Перетащите элемент **Порт** и дайте ему имя **plant_to_storage**. Соедините его с элементом **MoveTo** (рис. 5.20).

Этап 4. Моделирование агента Storage_Production

Моделирование входа в агент из агента Plant

Перейдите в агент **Storage_production** и перетащите на рабочее поле значок **Магазин** из библиотеки **Картинки** (рис. 5.21).

Перетащите элемент **Порт** из агентной библиотеки на рабочее поле агента **Storage_Production** и задайте его имя — **port_storage_production** (рис. 5.22).

Перейдите в агент **main** и соедините порты **plant_to_storage** и **port_storage_production**, как показано на рис. 5.23.

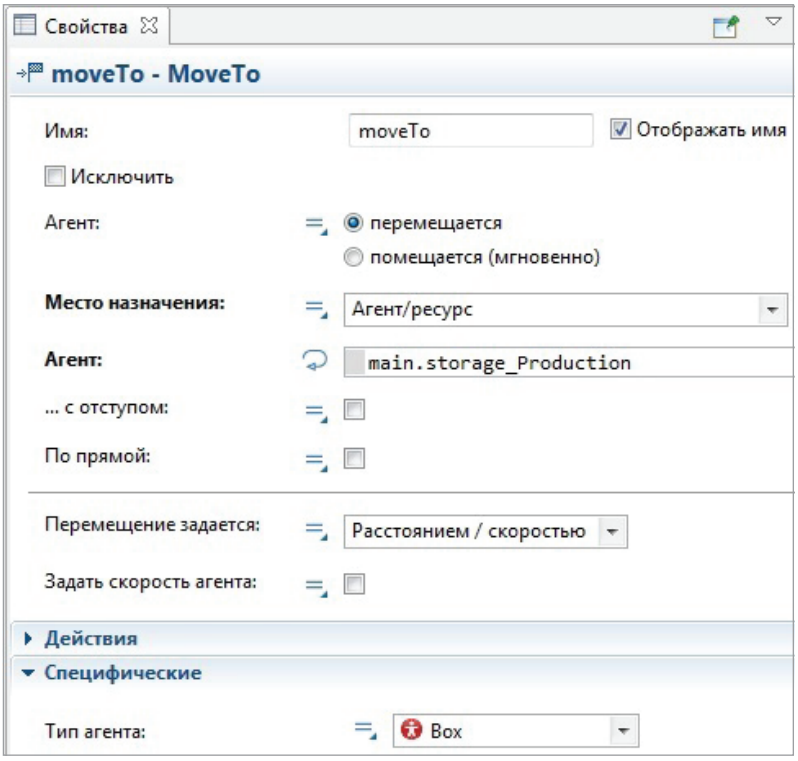

Рис. 5.19. Свойства элемента **MoveTo**

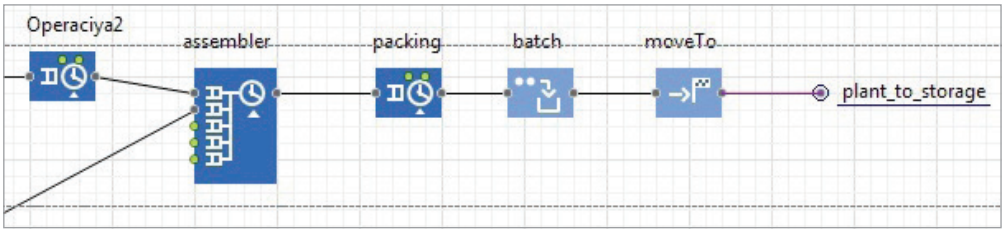

Рис. 5.20. Присоединение порта

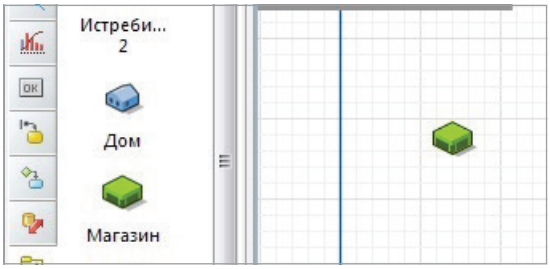

Рис. 5.21. Размещение картинки **Магазин**

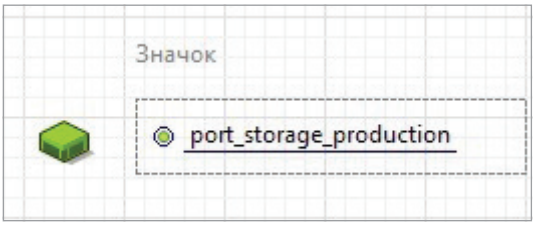

Рис. 5.22. Задание порта

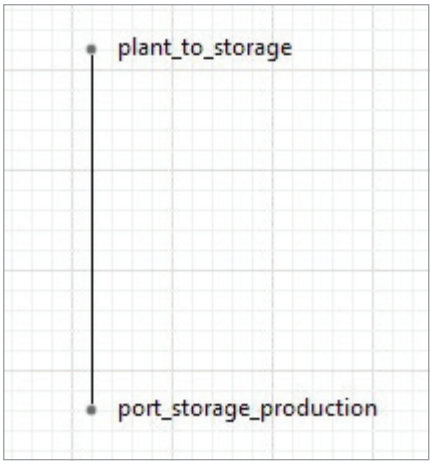

Рис. 5.23. Соединение портов на агенте **main**

Моделирование ухода коробок из модели

Перейдите в агент **Storage_production**, перетащите элемент **sink** из библиотеки моделирования процессов, соедините его с портом и задайте его свойства так, как показано на рис. 5.24.

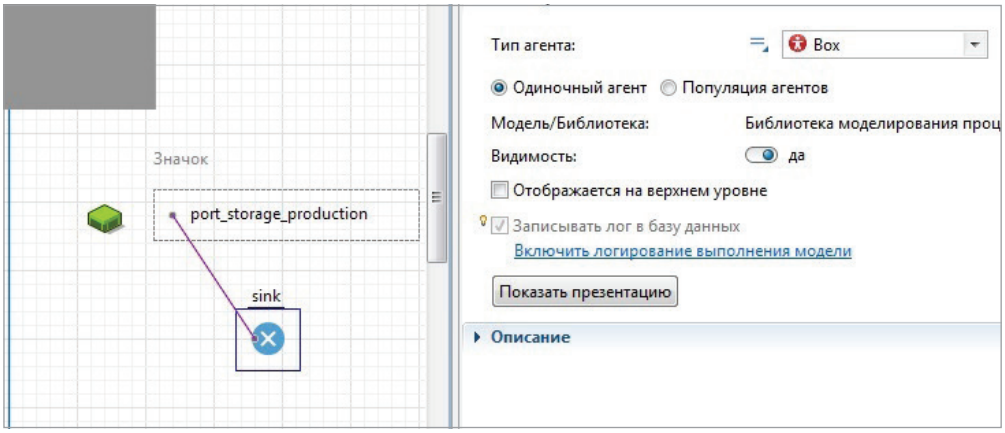

Рис. 5.24. Моделирование ухода коробок

Запустите модель, как на рис. 5.25–5.27.

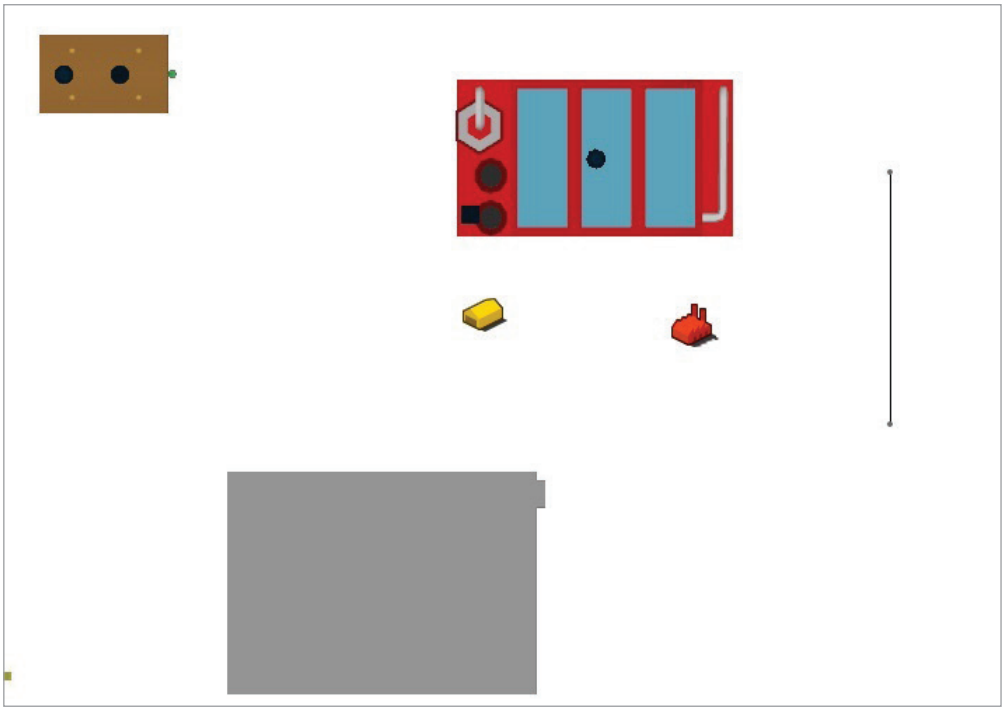

Рис. 5.25. Вид главного агента при запуске модели

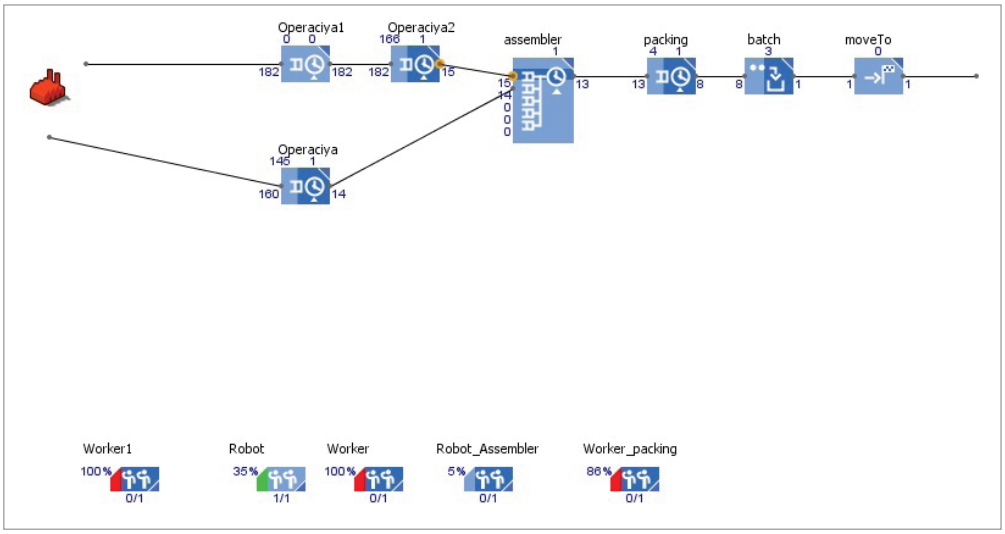

Рис. 5.26. Вид агента **Plant** при работе модели

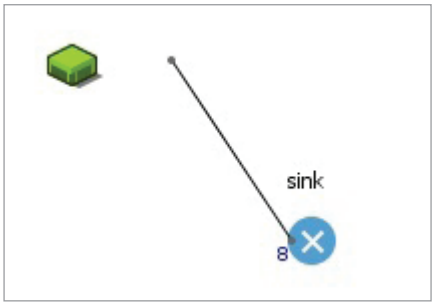

Рис. 5.27. Вид агента **Storage_Production** при работе модели

Лабораторная работа № 6

Создание смешанной агентно-дискретно-событийной модели

Задача

Суть этой работы сводится к тому, чтобы добавить производственный процесс к модели логистики, разработанной в лабораторной работе № 2.

Решение

Этап 1. Добавление агентов в модель

Откройте модель **Logistic** и перейдите в агент **main**. Добавьте в модель еще три единственных агента — агент **Detal**, агент **Detal2**, агент **Izdeliye**. Должно получиться так же, как на рис. 6.1.

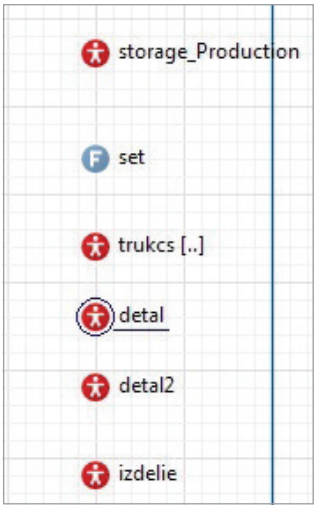

Рис. 6.1. Список агентов модели

Этап 2. Моделирование производства деталей на складе

Перейдите в агент **Storage**. Удалите в нем все содержимое, кроме анимации. В этой работе производство деталей будет промоделировано с помощью дискретно-событийного подхода.

Перетащите на рабочее поле агента элемент **Source** из библиотеки моделирования процессов и задайте его свойства, как показано на рис. 6.2.

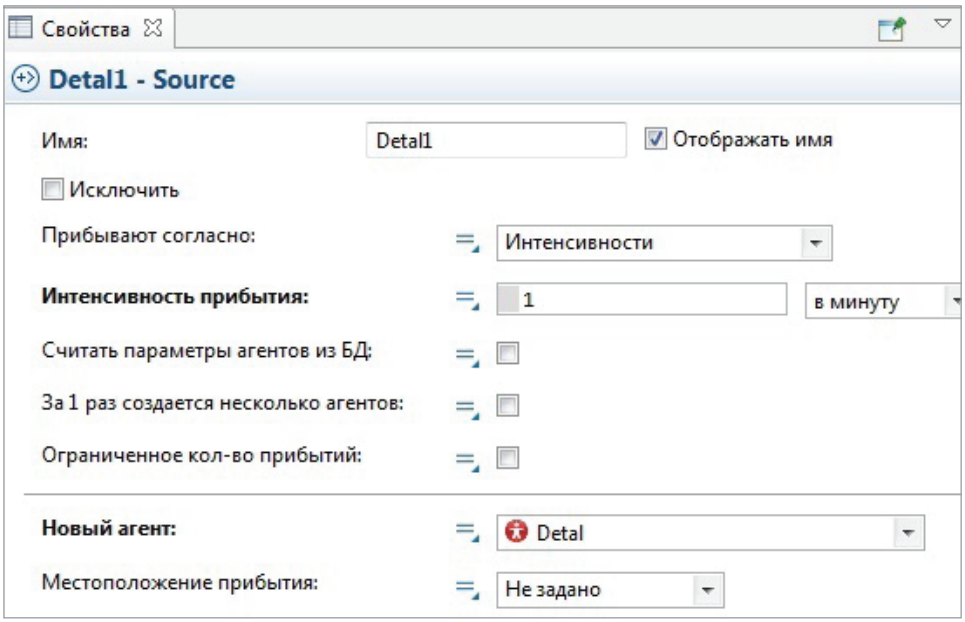

Рис. 6.2. Свойства источника деталей первого типа

Повторите операцию для моделирования производства деталей второго типа (рис. 6.3).

Этап 3. Организация связи между дискретно-событийной и агентной частью модели

Поскольку дискретно-событийный подход будет использован только для моделирования производственных процессов, а для моделирования логистики между производственными участками используется агентный подход, нужно организовать связь между этими частями модели.

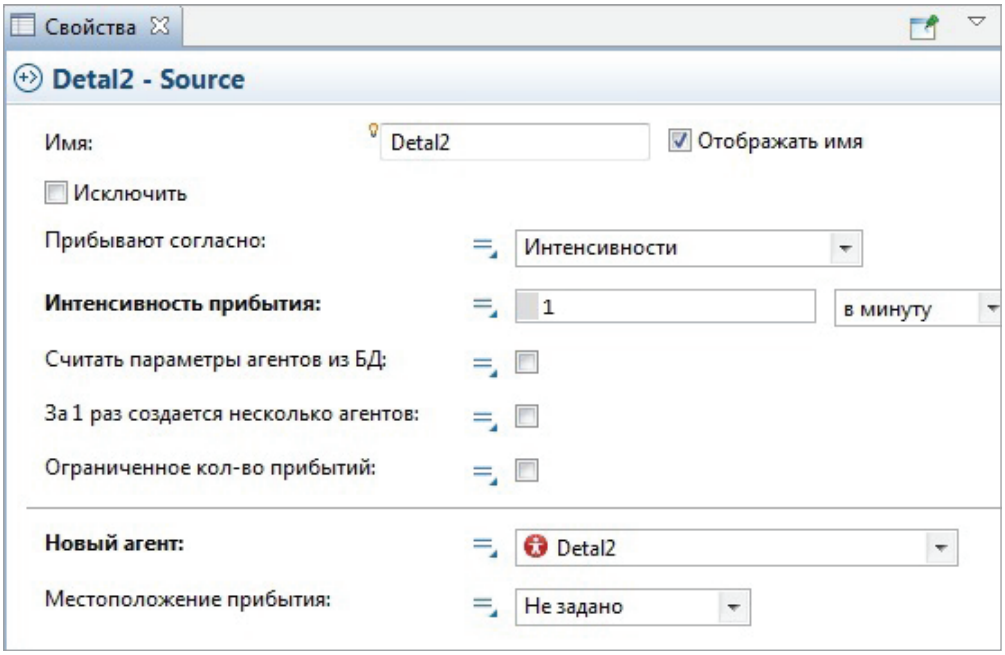

Рис. 6.3. Свойства источника деталей второго типа

Для связки дискретно-событийной части модели с агентной используется элемент **exit**, который превращает выходящую из дискретной части модели заявку в агента (рис. 6.4), и **enter**, который превращает входящего в дискретную часть модели агента в заявку (рис. 6.5).

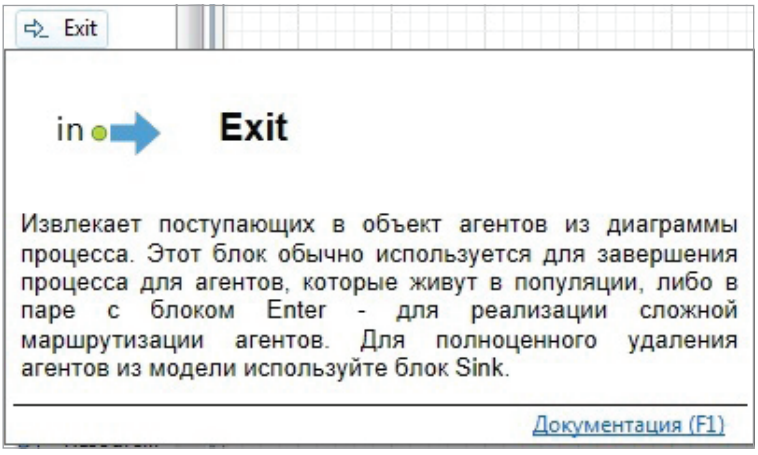

Рис. 6.4. Элемент **Exit**

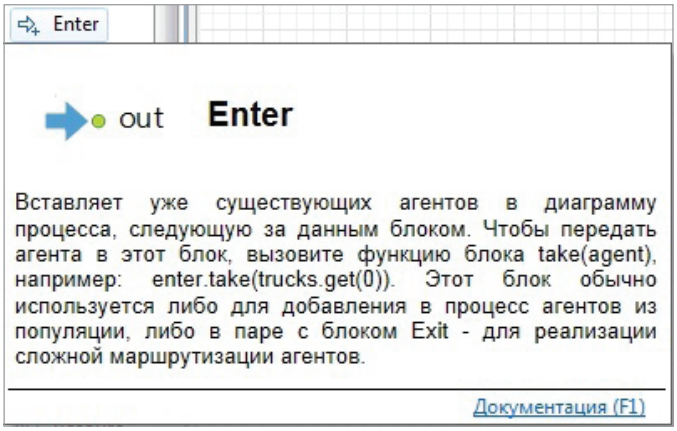

Рис. 6.5. Элемент **Enter**

Для организации выхода производимых в дискретной части модели деталей в агентную ее часть используется элемент **exit**. Перетащите этот элемент на рабочее поле, соедините его с элементом **Detal1** и дайте ему имя **exit_Detal1**. Повторите эту операцию для деталей второго типа (рис. 6.6).

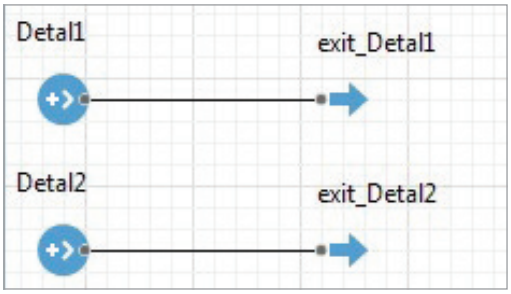

Рис. 6.6. Моделирование выходов деталей

Этап 4. Организация связи агентов Storage и Truck

Для того чтобы детали, производимые в агенте **Storage**, попали в грузовик, нужно связать агента **Storage** с агентом **Truck**. Для этого в агенте **Truck** нужно поместить элемент **enter** и связать его с элементом **exit** агента **Storage**.

Перейдите в агент **Truck**, перетащите элемент **enter** на рабочее поле агента и задайте ему имя **enter_detal1**. В свойствах элемента **enter_detal1** в разделе **Специфические** задайте тип агента **Detal**.

Повторите операцию и создайте вход **enter_detal2** с типом агента **Detal2**.

Перетащите значок **Грузовик** из библиотеки **Картинки** на рабочее поле агента **Truck**. Для связи с агентами нужен элемент **Порт** из агентной библиотеки. Перетащите элемент **Порт** из агентной библиотеки и дайте ему имя **port_Detal1_truck**. Повторите операцию и создайте порт **port_Detal2_truck** (рис. 6.7).

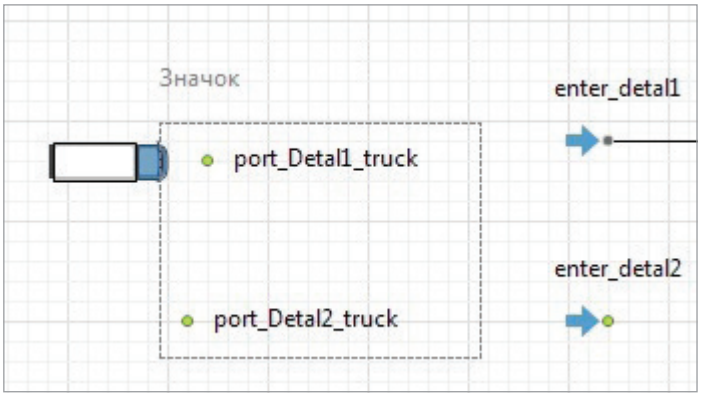

Рис. 6.7. Добавление портов для деталей второго типа

Перейдите в агент **Storage** и поместите в нем значок **Склад** из библиотеки **Картинки**, а также 2 элемента **Порт** из агентной библиотеки. Назовите их элементы — **port_Detal1** и **port_Detal2**. Соедините их с элементами **Detal1** и **Detal2** (рис. 6.8).

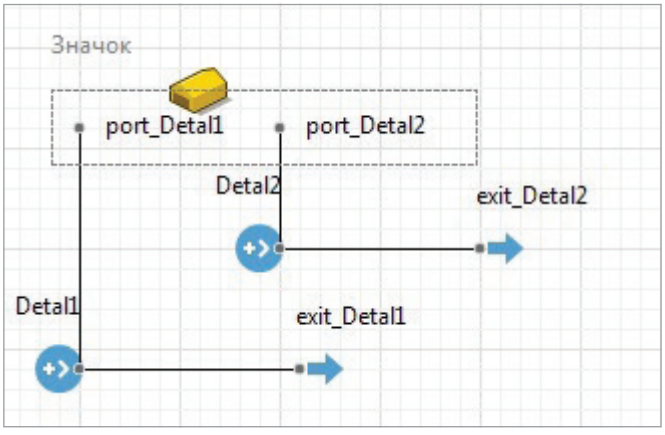

Рис. 6.8. Соединение портов в агенте **Storage**

Перейдите в агент **main**. Соедините порты так, как показано на рис. 6.9.

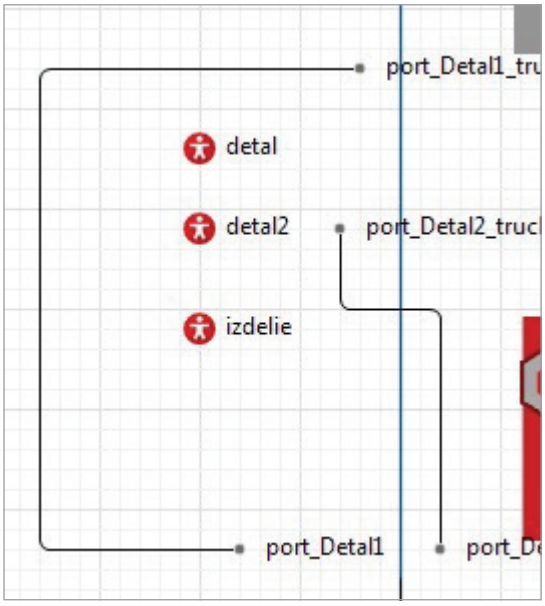

Рис. 6.9. Соединение портов в агенте **main**

Если мешают двойные надписи, то выделите порт и зайдите в свойства. В них уберите галочку с пункта **Отображать имя** (рис. 6.10).

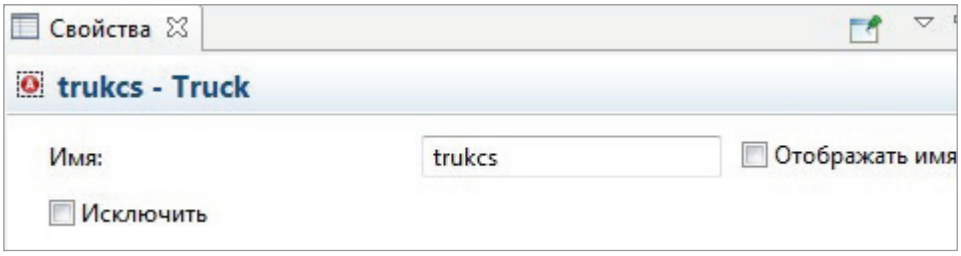

Рис. 6.10. Убрать двойные подписи

Далее, в выходных портах агента **Storage** нужно прописать, к какому входу должны уйти агенты.

Перейдите в агент **Storage**.

Агенты (детали) должны попасть в грузовик, который находится на складе, т. е. имеет состояние **atStorage**. Поэтому нужно сначала най-

ти грузовик в состоянии **atStorage**, а затем отправить агентов-детали в него. Для поиска грузовика, находящегося на складе, нужно написать функцию.

Перетащите элемент **Функция** из агентной библиотеки на рабочее поле агента **Storage** и задайте его свойства так, как показано на рис. 6.11.

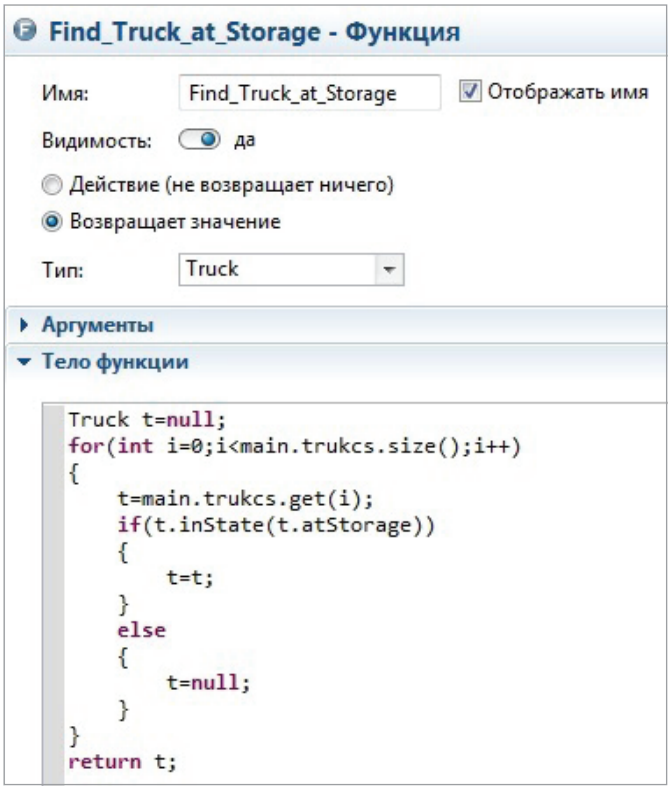

Рис. 6.11. Функция **Find_Truck_at_Storage**

Функция имеет имя **Find_Truck_at_Storage** и возвращает грузовик. В теле функции в первой строке объявляется переменная типа грузовик (**Truck**), в которую будет записан найденный грузовик. Пока грузовик не найден, в эту переменную записано значение **null**. В следующей строке начинается цикл, который проходит по всем грузовикам коллекции **trucks**, находящейся в главном агенте (агент **main**). Поэтому счетчик цикла меняется от **0** до размеров коллекции **trucks**. Размер коллекции дает функция **size()**, которая является стандартной (уже написана разработчиками **AnyLogic**) для коллекций. Поскольку кол-

лекция **trucks** находится в главном агенте, то сначала идет обращение к агенту **main**, потом через точку обращаемся к коллекции, а потом через точку обращаемся к функции коллекции. В первой строке цикла в переменную **t** записываем очередной грузовик из коллекции. Грузовик получаем с помощью стандартной функции **get()**, которая возвращает элемент коллекции с указанным в скобках индексом. Далее проверяем, находится ли грузовик в состоянии **atStorage**: если да, то это нужный нам грузовик и мы его записываем в переменную **t**; если нет, то в переменную **t** записываем **null**. В конце функции возвращаем найденный грузовик.

Теперь, имея функцию, можно отправить детали в нужный грузовик.

Выделите выход **exit_Detal1** и задайте в его свойствах действие, которое отправит детали первого типа во вход для них в агенте **Truck**, как это показано на рис. 6.12.

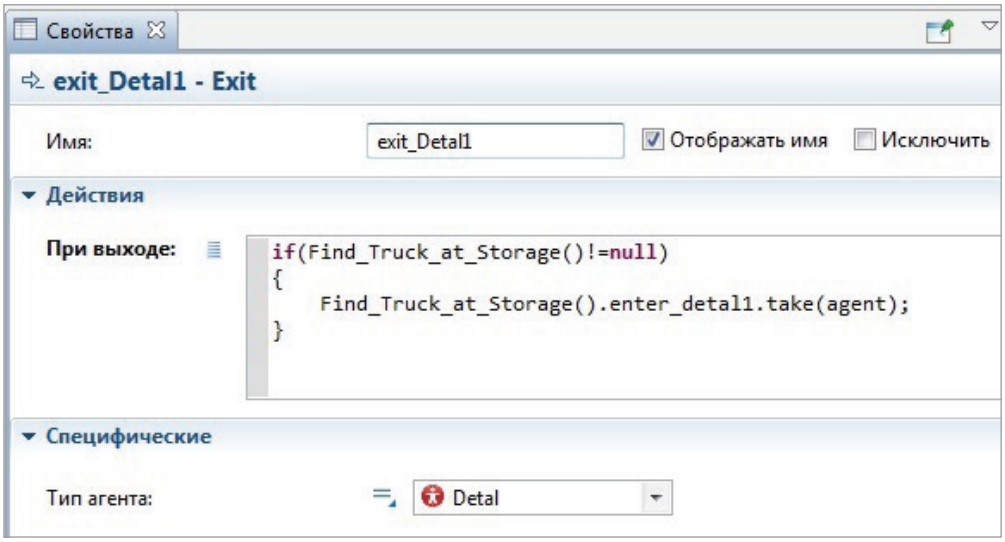

Рис. 6.12. Свойства выхода **exit_Detal1**

В действиях на выходе сначала проверяется, нашелся ли грузовик (если функция **Find_Truck_at_Storage()** что-то вернула), затем во вход найденного грузовика отправляется деталь с помощью стандартной функции **take()**.

Теперь повторите операцию для выхода **exit_Detal2** и добавьте в его свойства действие, показанное на рис. 6.13.

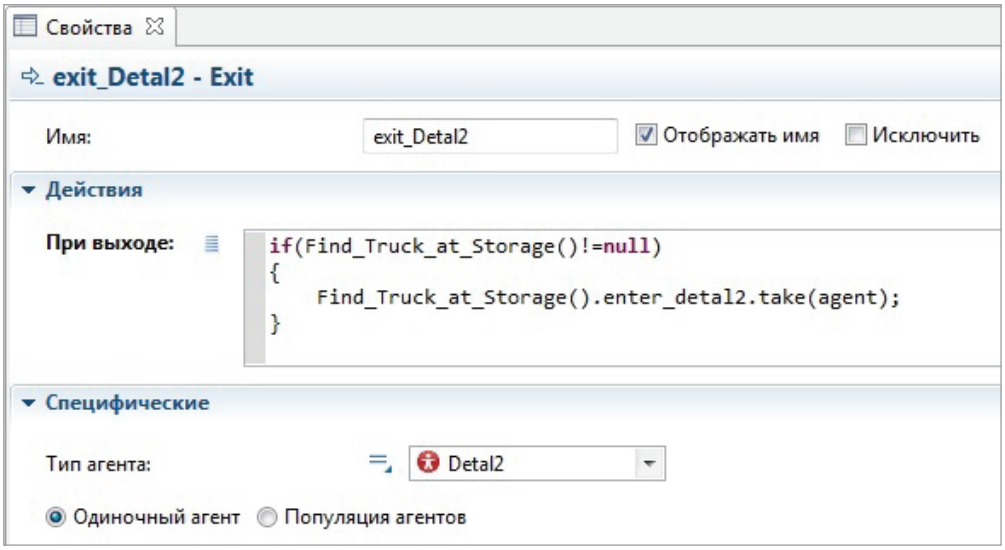

Рис. 6.13. Добавление действия к элементу **exit2**

Этап 5. Моделирование операции погрузки (разгрузки) деталей в агенте Truck

Перейдите в агент **Truck**.

Операцию погрузки (разгрузки) промоделируем с помощью элемента **Service** библиотеки моделирования процессов.

Перетащите элемент **Service** на рабочее поле агента **Truck** и соедините его со входом **enter_Detal1**. Повторите эту операцию для входа **enter_Detal2** (рис. 6.14).

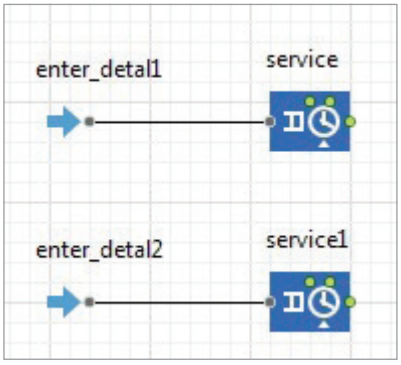

Рис. 6.14. Соединение выходов в агенте **Truck**

Задайте свойства блоков **Service**, как показано на рис. 6.15. Погрузка выполняется от 5 до 12 минут (в среднем 7) одним рабочим, который производит погрузку как деталей первого типа, так и деталей второго типа. Он может передвигаться между операциями, поэтому для него выбран тип ресурса **Движущийся** (рис. 6.16). Рабочий работает по расписанию с 8 до 17 ч с перерывом на обед с 12 до 13.

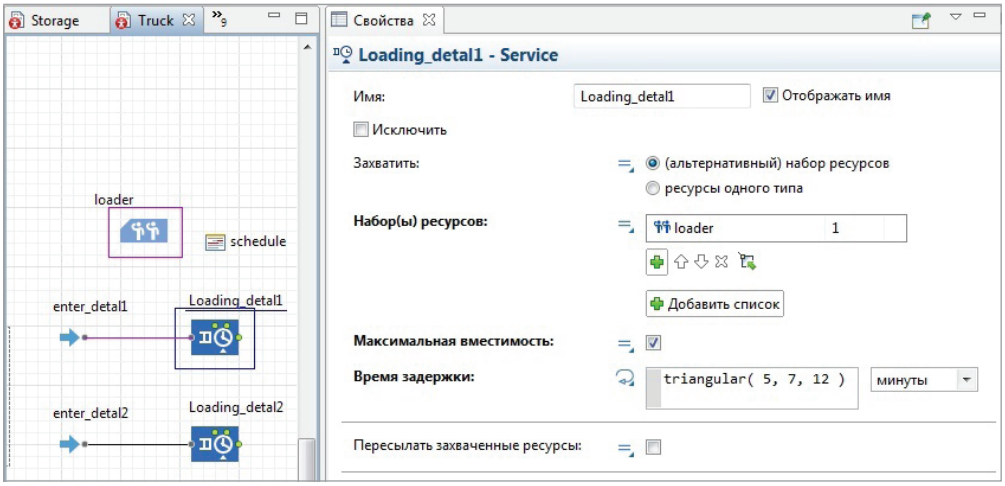

Рис. 6.15. Свойства операции **Loading**

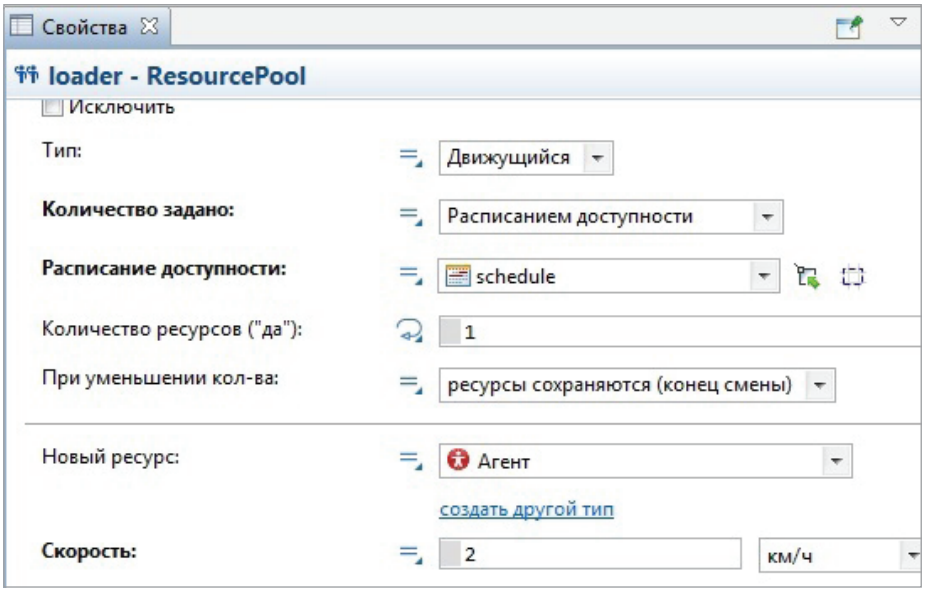

Рис. 6.16. Свойства ресурсов

Этап 6. Организация связи агентов Truck и Plant

Для того чтобы детали из грузовиков попали в цех, нужно связать агентов **Truck** и **Plant**.

Для этого в агент **Truck** нужно поместить выходы и связать их со входами, помещенными в агент **Plant**.

Перетащите на рабочее поле агента **Truck2** элемента **exit** из библиотеки моделирования процессов и соедините их так, как показано на рис. 6.17.

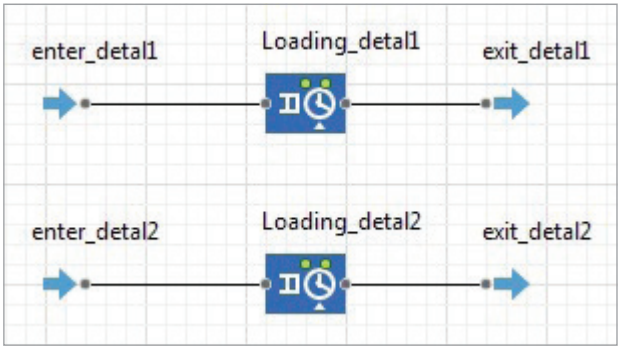

Рис. 6.17. Соединение портов в агенте **Truck2**

Перейдите в агент **Plant** и уберите все лишнее. Должны остаться функции **Find_Truck, detal_deliver**, **izdeliya** и события, которые вызывают функции **detal_deliver** и **izdeliya**.

Перетащите на рабочее поле агента значок **Завод** из библиотеки **Картинки** (рис. 6.18).

Перетащите два элемента **Порт** из агентной библиотеки и дайте им имена **port_detal1_plant** и **port_detal2_plant**.

Перейдите в агент **main** и соедините порты так, как показано на рис. 6.19.

Вернитесь в агент **Plant** и перетащите на рабочее поле агента 2 элемента **enter**. Назовите их **enter_Detal1** и **enter_Detal2**.

Перейдите в агент **Truck**, выделите выход **exit_Detal1** и в его свойствах укажите, к какому входу его привязать (рис. 6.20).

Аналогично привяжите выход **exit_Detal2** (рис. 6.21).

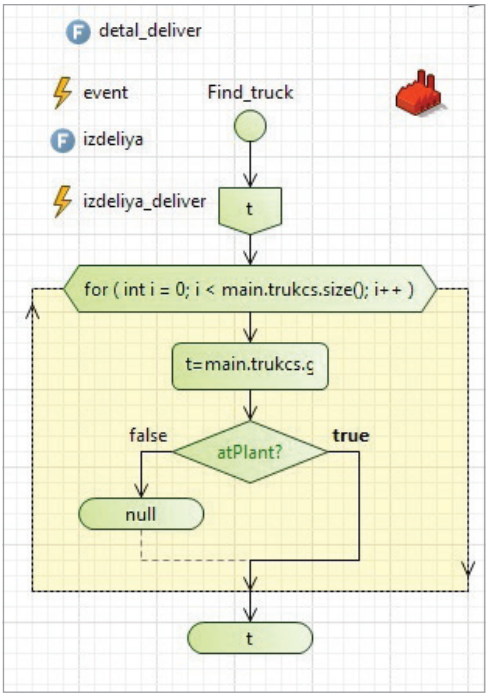

Рис. 6.18. Размещение значка **Завод** в агенте **Plant**

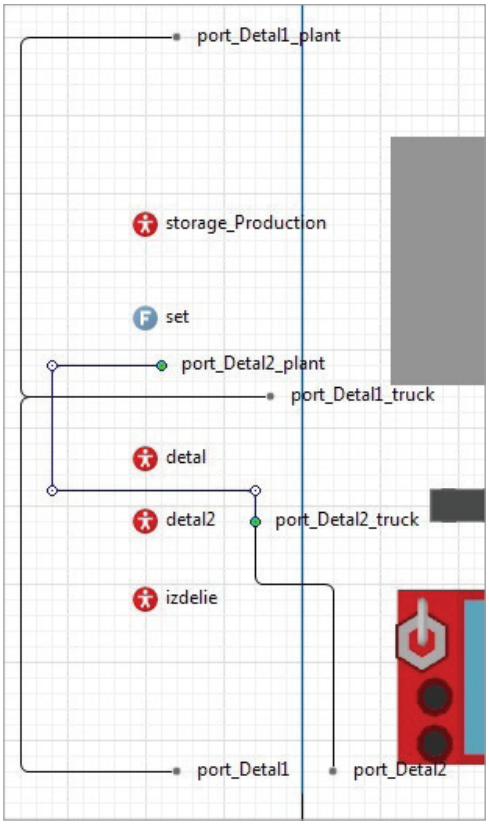

Рис. 6.19. Соединение портов в агенте **main**

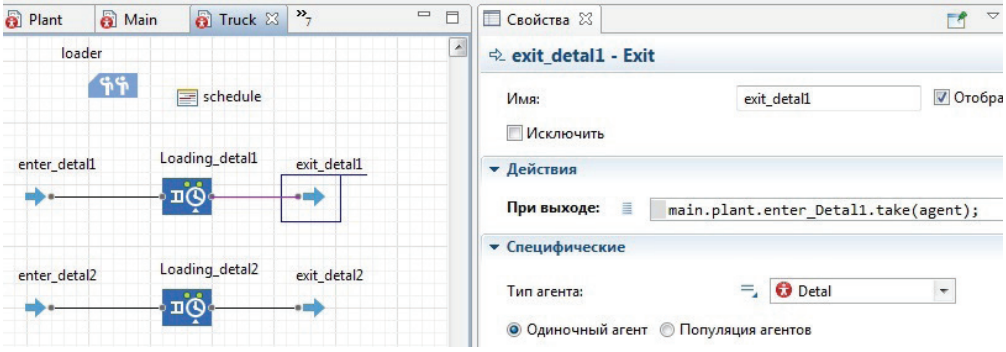

Рис. 6.20. Привязка выходов агента **Detal1**

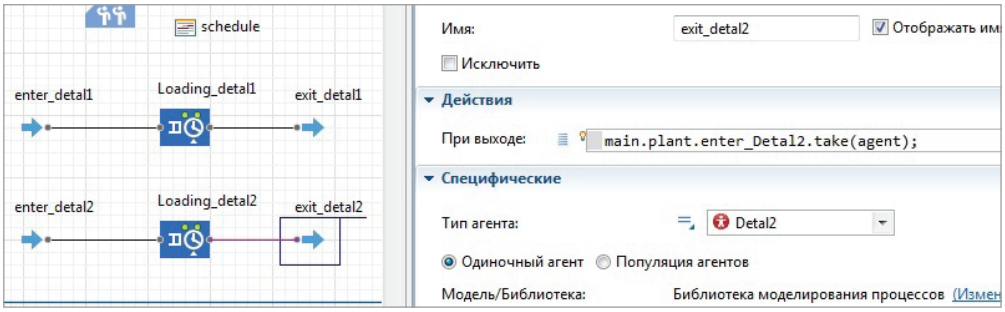

Рис. 6.21. Привязка выходов агента **Detal2**

Этап 7. Ликвидация всего лишнего из агента Truck

В агенте **Truck** есть диаграмма состояний. В ней к каждому переходу привязано действие. Ранее в каждом переходе изменялось значение параметра **order**, теперь этот параметр не нужен (его можно удалить). Следовательно, нужно убрать все, что с ним связано из действий в переходах. В результате должно остаться:

- в переходе между состояниями **atPlant** и **atStorage**, как на рис. 6.22;

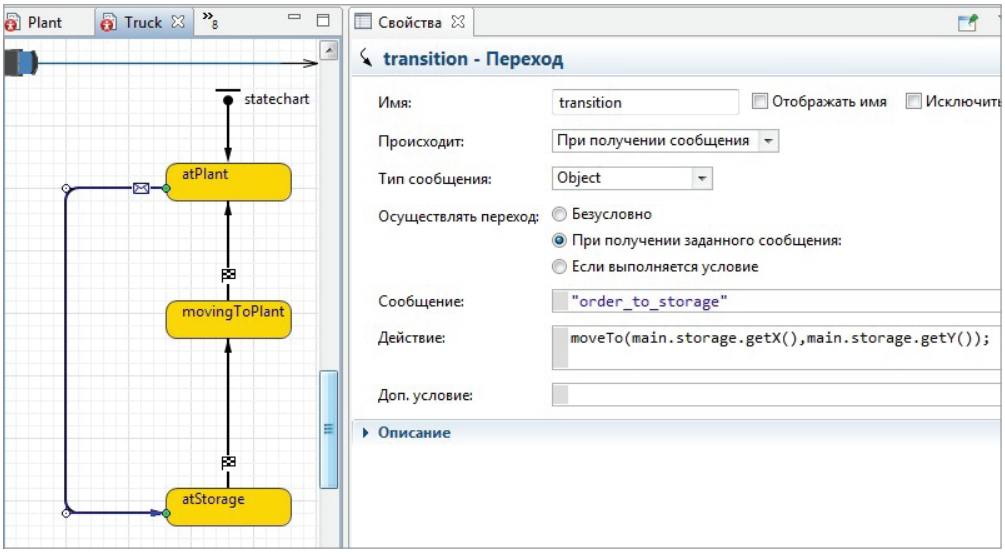

Рис. 6.22. Задание перехода между состояниями **atPlant** и **atStorage** агента **Truck**

- в переходе между состояниями **atStorage** и **movingToPlant**, как на рис. 6.23;

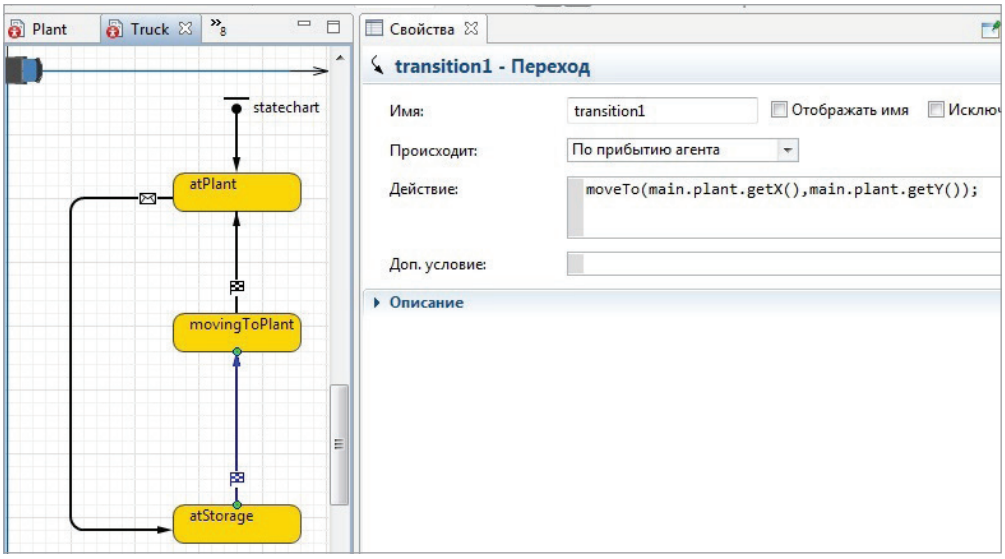

Рис. 6.23. Задание перехода между состояниями **atStorage** и **movingToPlant** агента **Truck**

- в переходе между состояниями **movingToPlant** и **atPlant**, как на рис. 6.24.

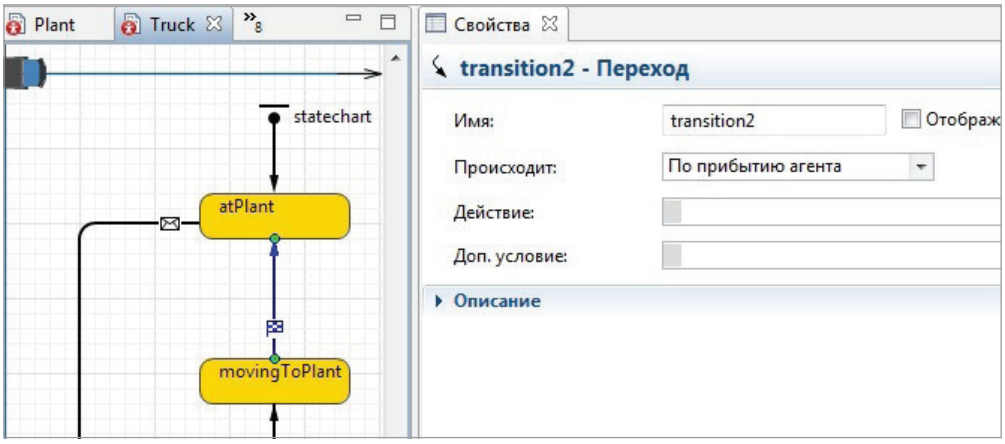

Рис. 6.24. Задание перехода между состояниями **movingToPlant** и **atPlant** агента **Truck**

Этап 8. Моделирование производственного процесса в цехе

Перейдите в агент **Plant**. Производственный процесс сборки промоделируем с помощью элементов библиотеки моделирования процессов точно так же, как и в предыдущей работе. К входам **enter_Detal1** и **enter_Detal2** присоедините элементы **Service** и задайте их свойства, как показано на рис. 6.25 и 6.26.

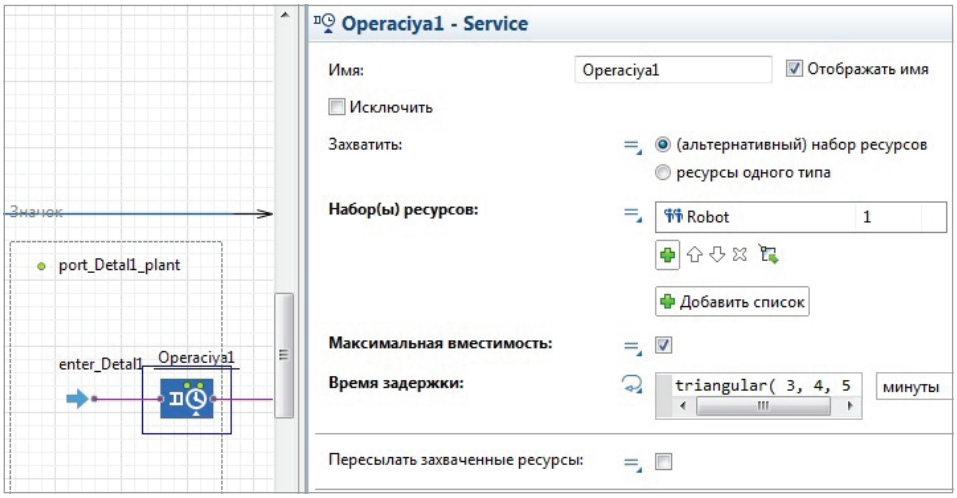

Рис. 6.25. Задание свойств **Operaciya1**

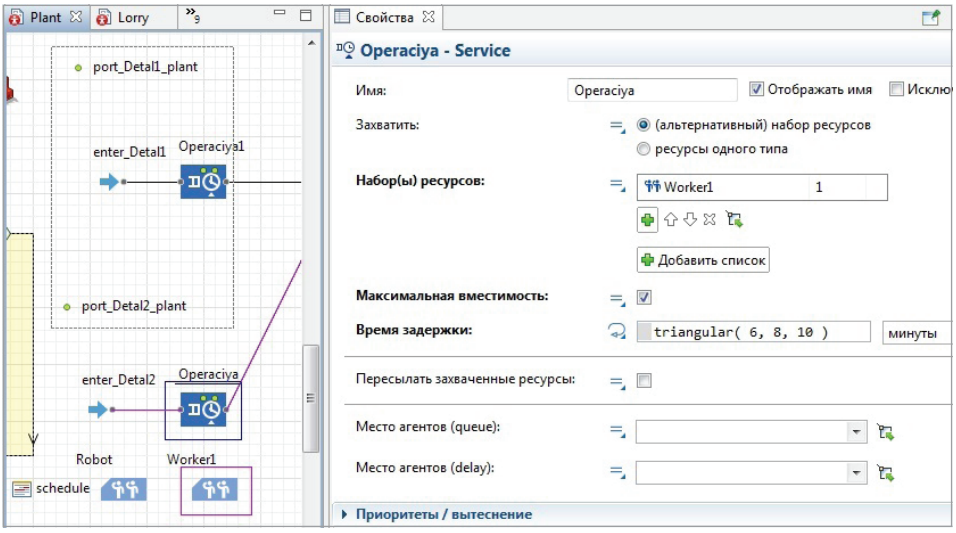

Рис. 6.26. Задание свойств **Operaciya**

Далее, повторите все действия, как в предыдущей работе, за исключением операции упаковки. Конечный результат представлен на рис. 6.27.

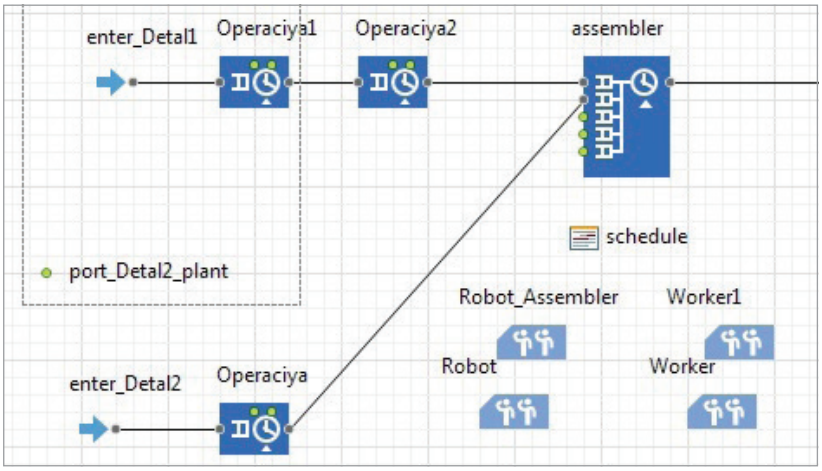

Рис. 6.27. Конечный результат

Этап 9. Изменение условий отправки грузовиков за деталями на склад

Грузовик должен отправляться на склад за деталями, если их осталось меньше 2. Количество оставшихся деталей равно разнице поступивших деталей и потраченных на сборку. Для этого расчета введем переменные **Detal1_in_plant** и **Detal2_in_plant**. Для задания переменной используется элемент **Переменная** агентной библиотеки. Перетащите 2 этих элемента и задайте их свойства, как показано на рис. 6.28 (свойства у них одинаковые, отличаются только имена).

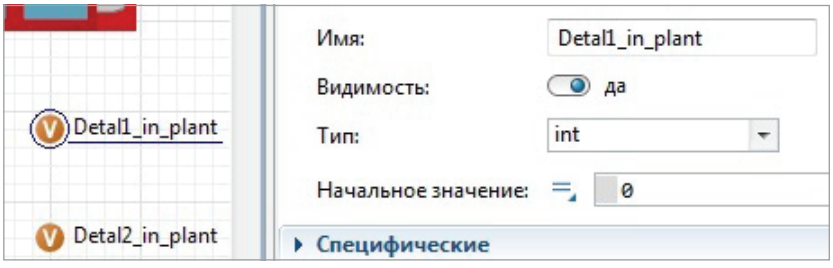

Рис. 6.28. Задание свойств переменных

Выделите элемент **enter_Detal1** и в его свойствах в разделе **Действия** задайте увеличение переменной **Detal1_in_plant** на 1 при входе в него детали из грузовика, как показано на рис. 6.29.

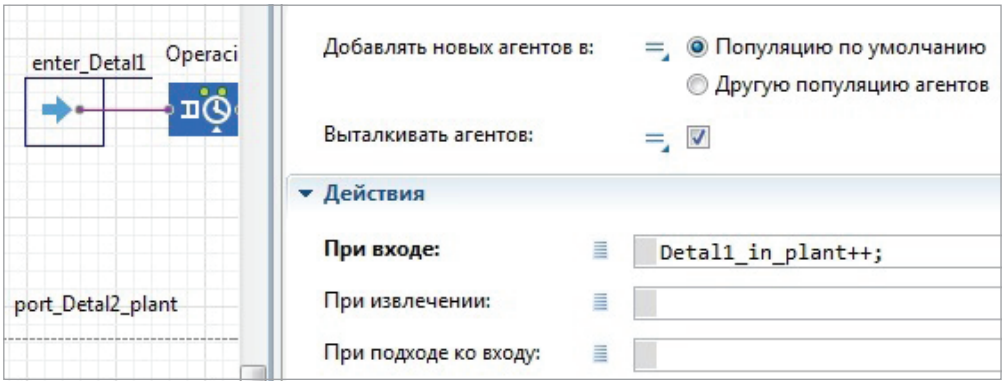

Рис. 6.29. Задание увеличения количества деталей первого типа

Повторите эту операцию для входа **enter_Detal2** (рис. 6.30).

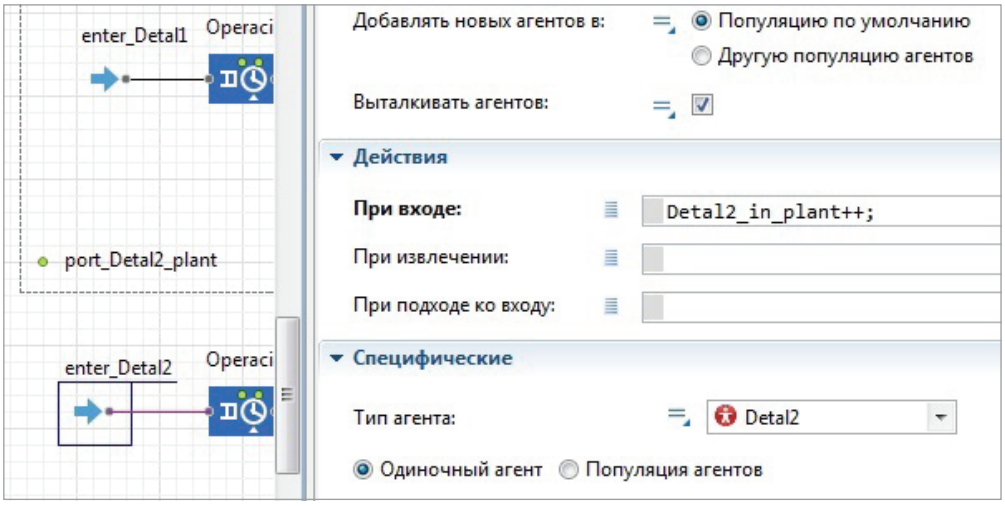

Рис. 6.30. Задание увеличения количества деталей второго типа

Выделите элемент **Operaciya2** и в его свойствах в разделе действия уменьшите количество деталей на 1, как показано на рис. 6.31.

Повторите это действие для элемента **Operaciya** (рис. 6.32).

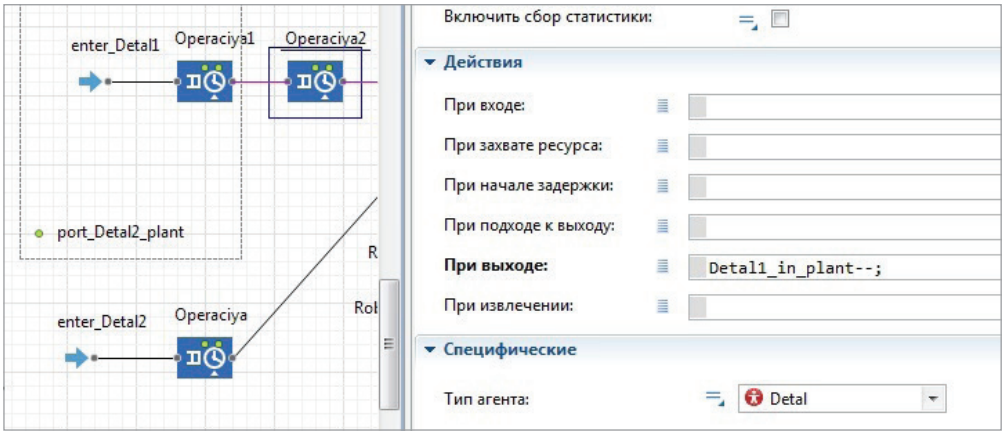

Рис. 6.31. Задание уменьшения количества деталей первого типа

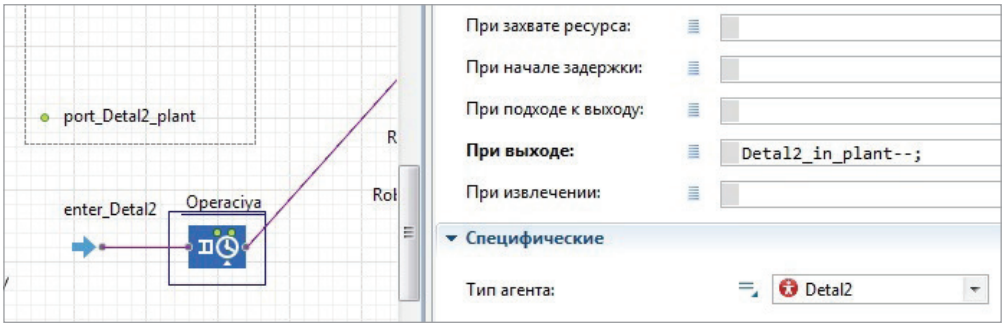

Рис. 6.32. Задание уменьшения количества деталей второго типа

Выделите функцию **detal_deliver** и в ее теле добавьте в условия проверку переменных **Detal1_in_plant** и **Detal2_in_plant**, как показано на рис. 6.33.

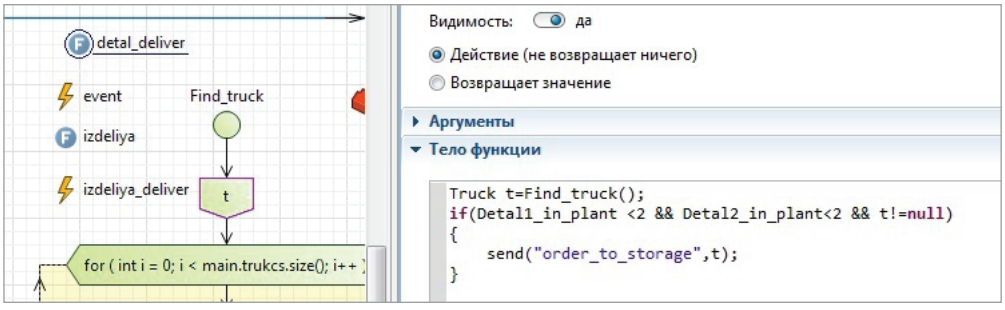

Рис. 6.33. Добавление в функцию **detal_deliver** условия проверки значений переменных

Этап 10. Организация связи агентов Plant и Lorry

Для доставки готовых изделий на склад используется грузовик другого типа — агент **Lorry**. Для того чтобы изделия попадали в грузовик, нужно организовать связь агентов **Plant** и **Lorry**. Для этого в конце производственной цепочки нужно поместить элемент **exit** и связать его со входом в агенте **Lorry**. Перетащите элемент **exit** в конец цепочки и назовите его **exit_Izdelia_plant** (рис. 6.34).

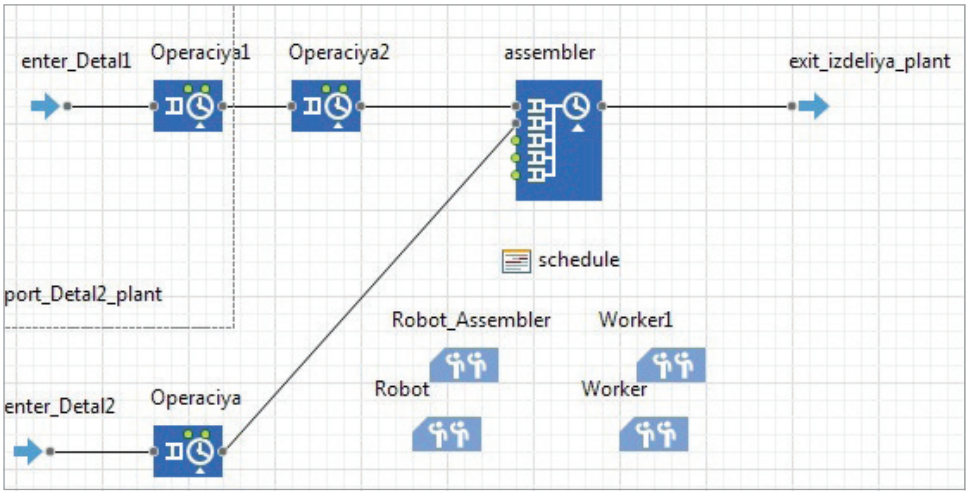

Рис. 6.34. Моделирование связи агентов **Plant** и **Lorry**

Перейдите в агент **Lorry** и перетащите значок **Грузовик** из библиотеки **Картинки**. Поместите элемент **Порт** из агентной библиотеки и назовите его **port_Izdeliya_Lorry** (рис. 3.35).

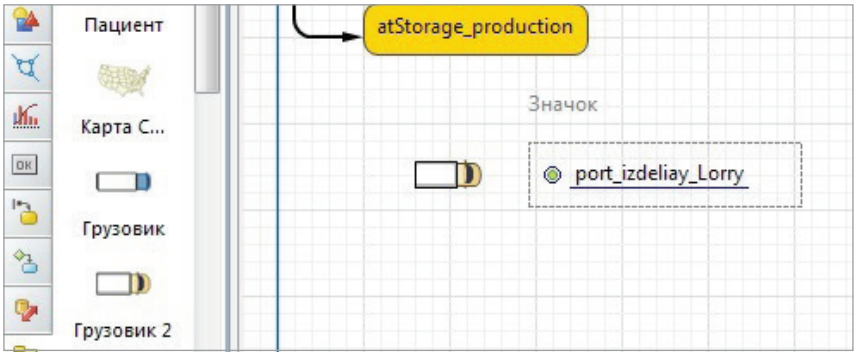

Рис. 6.35. Размещение картинки грузовика в агенте **Lorry**

Поместите на рабочее поле агента **Lorry** элемент **enter** из библиотеки моделирования процессов и назовите его **enter_izdeliya_lorry**, как показано на рис. 6.36.

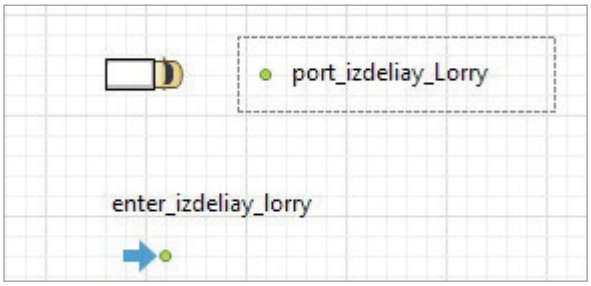

Рис. 6.36. Размещение входа в агент **Lorry**

Перейдите в агент **Plant**, поместите на его рабочее поле элемент **Порт** и назовите его **port_izdeliay_plant**.

Перейдите в агент **main** и соедините порты так, как показано на рис. 6.37.

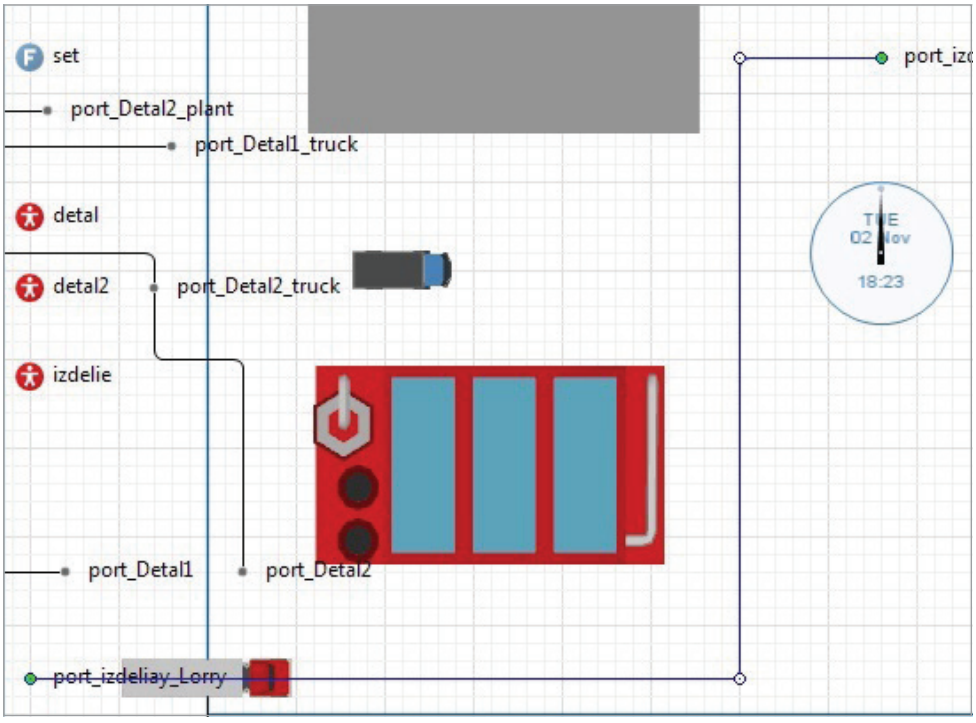

Рис. 6.37. Размещение порта в агенте **Plant**

Перейдите в агент **Lorry** и задайте тип агента **Izdelie** в элементе **enter_izdelia_lorry**, как показано на рис. 6.38.

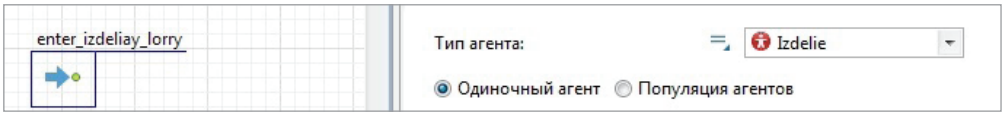

Рис. 6.38. Задание типа изделия в агенте **Izdelie**

Перейдите в агент **Plant** и зайдите в свойства элемента **exit_izdeliya_plant**. Задайте привязку этого выхода ко входу в агенте **Lorry**, как показано на рис. 6.39.

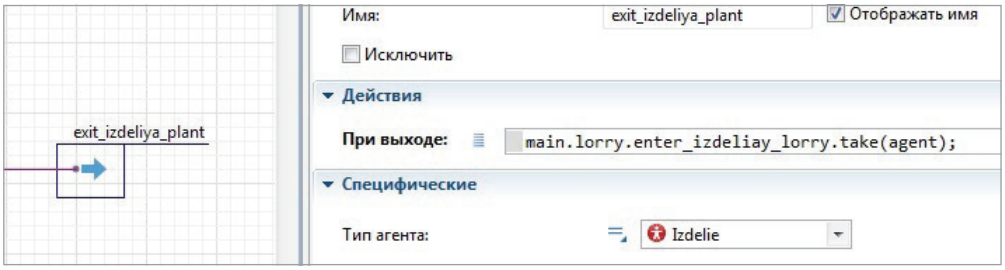

Рис. 6.39. Задание свойства элемента **exit_izdeliya_plant**

Этап 11. Ликвидация всего лишнего из агента Lorry

В агенте есть диаграмма состояний и параметр **order**. Параметр больше не понадобится, поэтому его нужно удалить. Также нужно удалить все действия в диаграмме состояния, связанные с этим параметром. В результате в диаграмме состояния должно остаться:

- в переходе из состояния atPlant в состояние atStorage_production, как на рис. 6.40;
- в переходе из состояния atStorage_production в состояние movingToPlant, как на рис. 6.41;
- в переходе из состояния movingToPlant в состояние atPlant в действиях ничего не должно остаться (рис. 6.42).

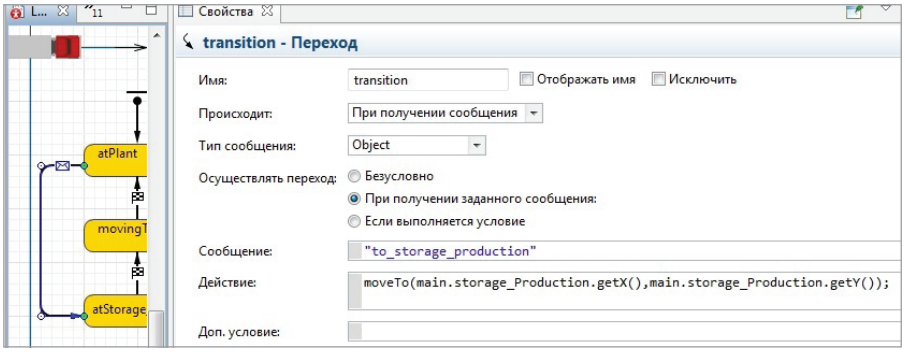

Рис. 6.40. Изменение перехода из состояния **atPlant** в состояние **atStorage_production** в агенте **Lorry**

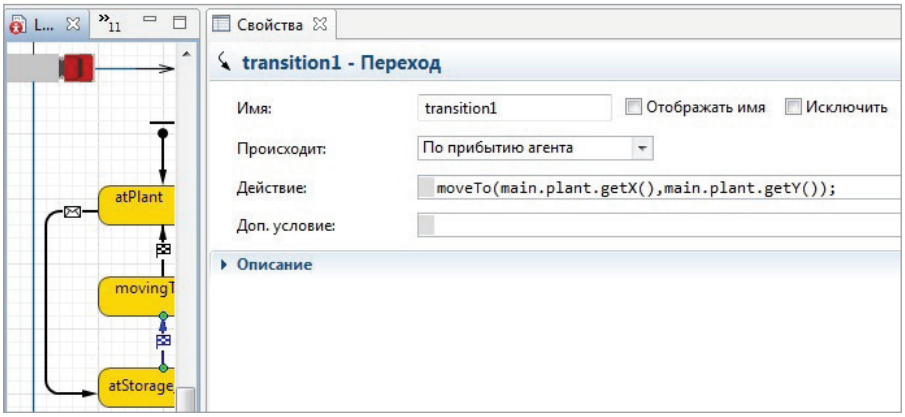

Рис. 6.41. Изменение перехода из состояния **atStorage_production** в состояние **movingToPlant** в агенте **Lorry**

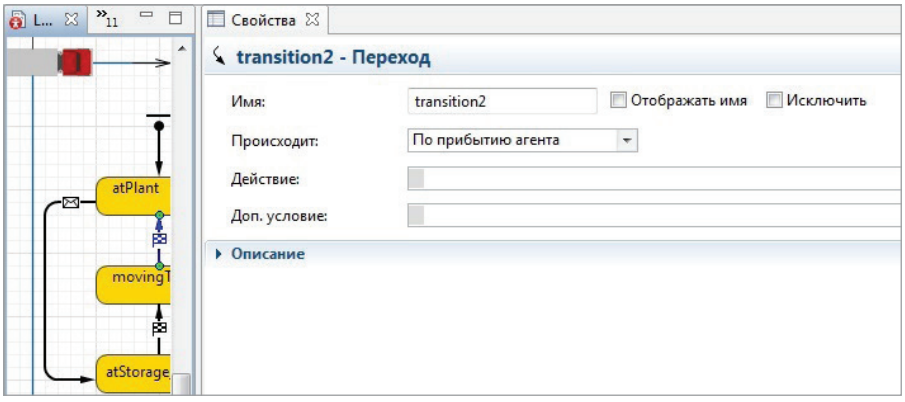

Рис. 6.42. Переход из состояния **movingToPlant** в состояние **atPlant** в агенте **Lorry**

Этап 12. Моделирование процесса разгрузки (погрузки) в агенте Lorry

Перейдите в агент **Lorry**. Процесс разгрузки (погрузки) промоделируем элементом **Service**.

Перетащите элемент на рабочее поле агента **Lorry** и задайте его свойства так, как показано на рис. 6.43.

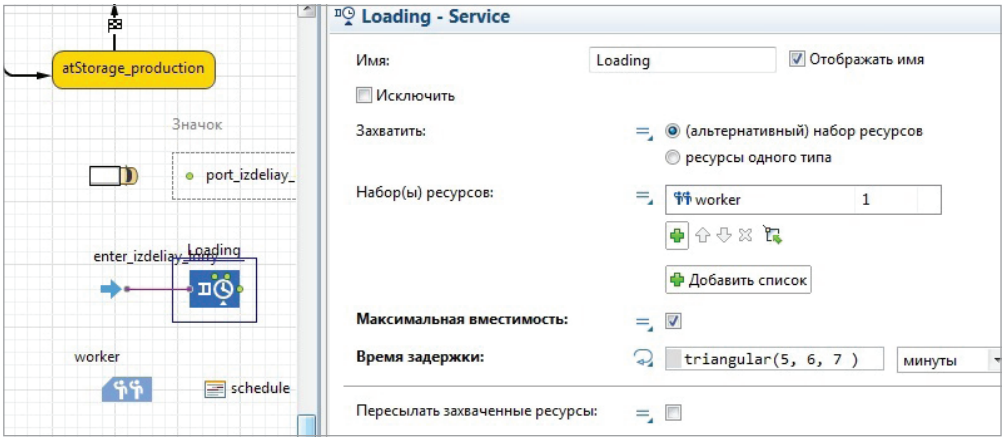

Рис. 6.43. Задание свойств операции **Loading**

Этап 13. Организация связи агента Lorry и агента Storage_Production

Для того чтобы изделия попали на склад нужно связать агентов **Lorry** и **Storage_Production**. Для этого в агент **Lorry** нужно поместить элемент **exit** (рис. 6.44).

Перейдите в агент **Storage_Production** и поместите в нем порт и значок **Магазин**, как показано на рис. 6.45.

Перейдите в агент **main** и соедините порты, как показано на рис. 6.46.

Поместите элемент enter в агент **Storage_Production**, в его свойствах укажите тип агента **izdelie** и сразу поместите элемент **Sink** (рис. 6.47).

Перейдите в агент **Lorry** и свяжите выход со входом агента **Storage_Production** (рис. 6.48).

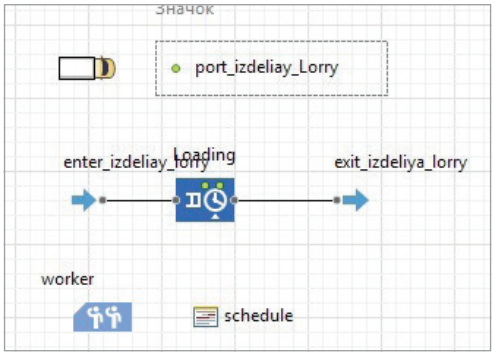

Рис. 6.44. Размещение выхода из агента **Lorry**

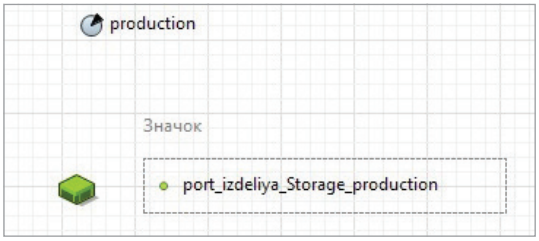

Рис. 6.45. Размещение порта в агенте **Strage_Production**

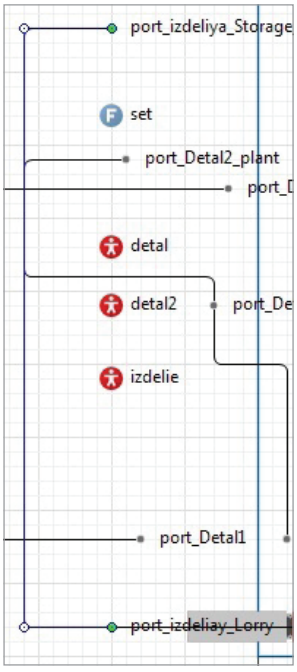

Рис. 6.46. Соединение портов в агенте **main**

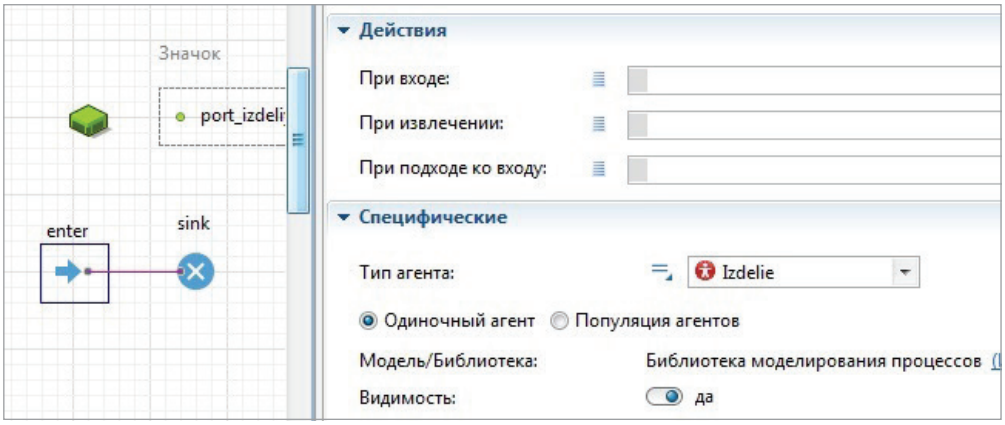

Рис. 6.47. Задание свойств элемента **enter** в агенте **Storage_Production**

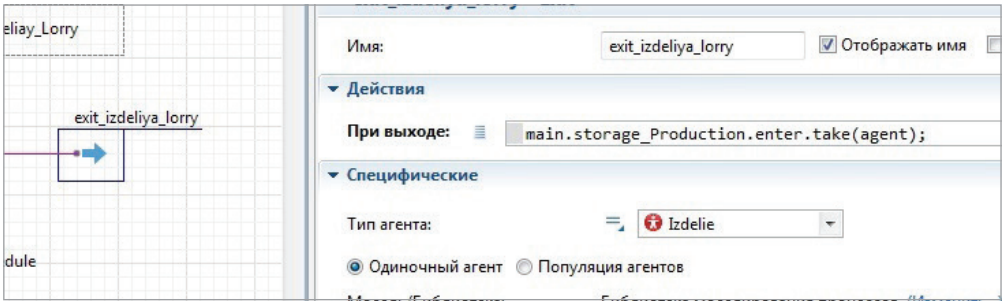

Рис. 6.48. Соединение входа и выходов в агенте **Lorry**

Этап 14. Изменение условий, при которых отправляются готовые изделия из цеха на склад готовых изделий

Перейдите в агент **Plant**.

Когда в цехе появляются готовые изделия, их надо отвезти на склад готовых изделий. Для подсчета готовых изделий нужно создать переменную. Перетащите элемент **Переменная** на рабочее поле агента **Plant** и задайте ее свойства, как показано на рис. 6.49.

Зададим увеличение этой переменной при окончании сборки, как это показано на рис. 6.50.

Теперь изменим условия в функции **izdeliya()**. Для этого зайдите в свойства функции и измените ее код, как показано на рис. 6.51.

Запустите модель (рис. 6.52–6.56).

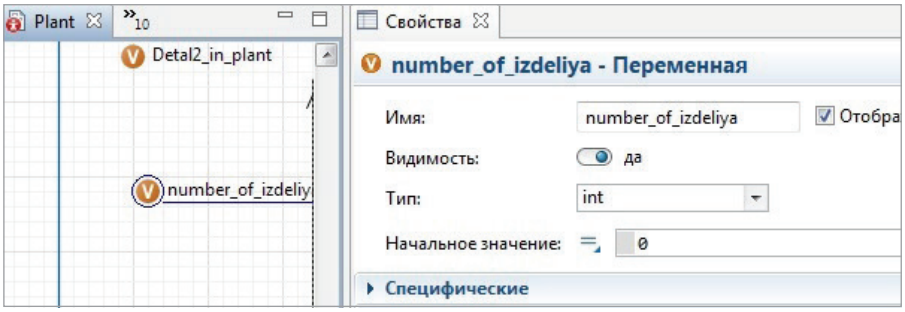

Рис. 6.49. Задание свойств переменной number_of_izdeliya

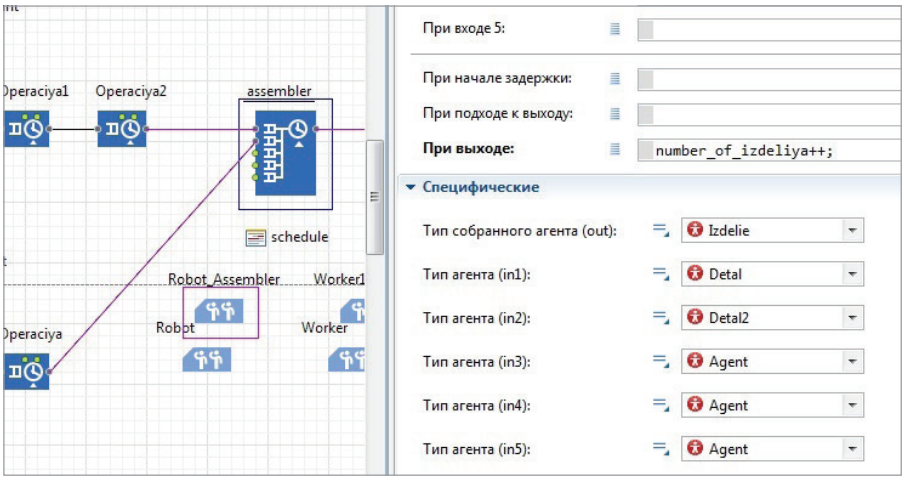

Рис. 6.50. Задание увеличения количества изделий

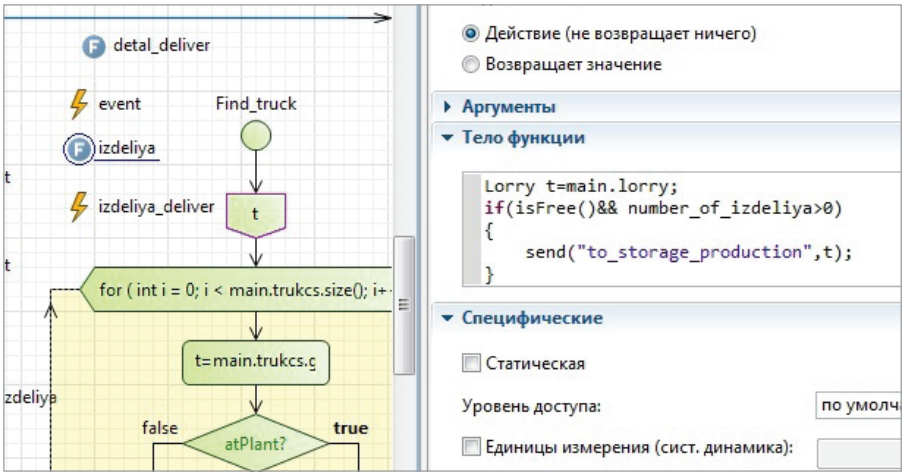

Рис. 6.51. Изменение условий в функции

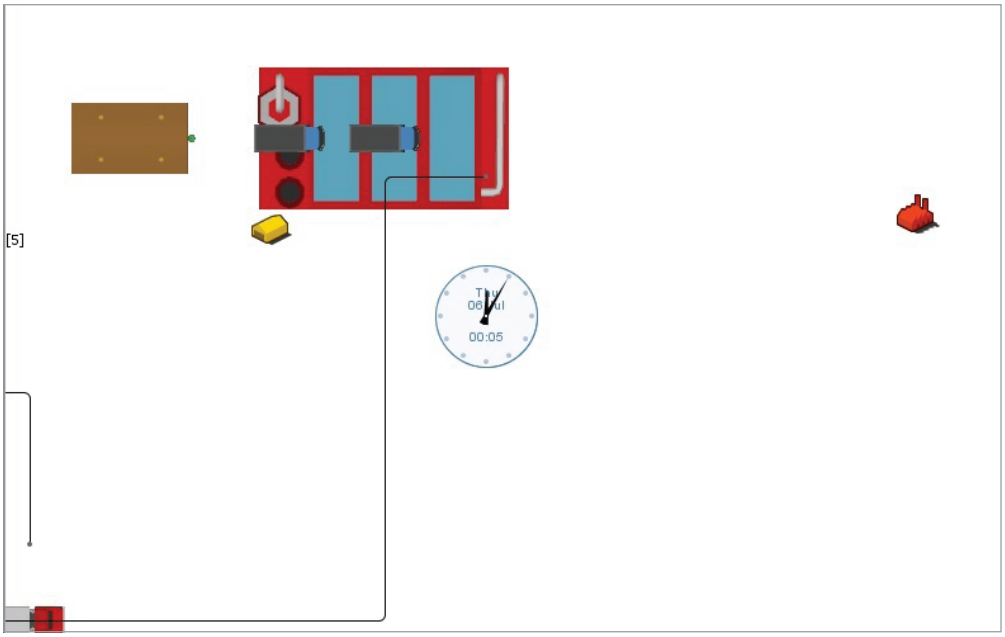

Рис. 6.52. Работа агента **Main**

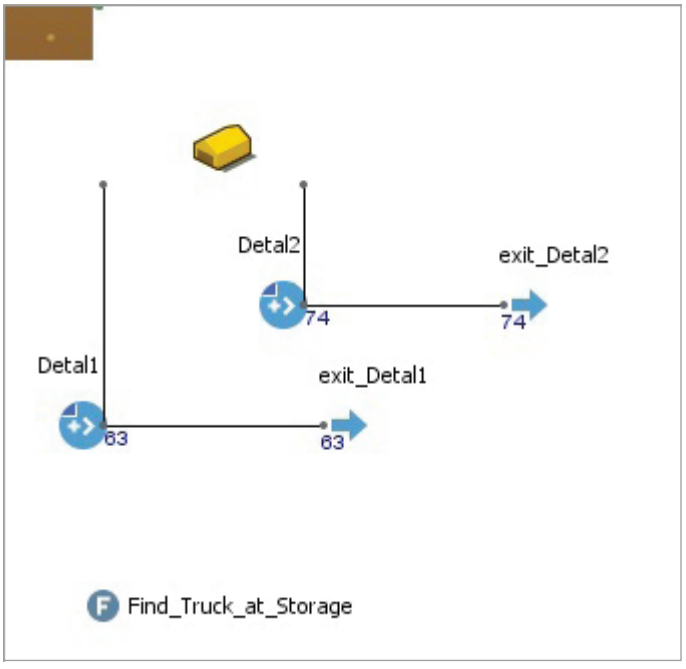

Рис. 6.53. Работа агента **Storage**

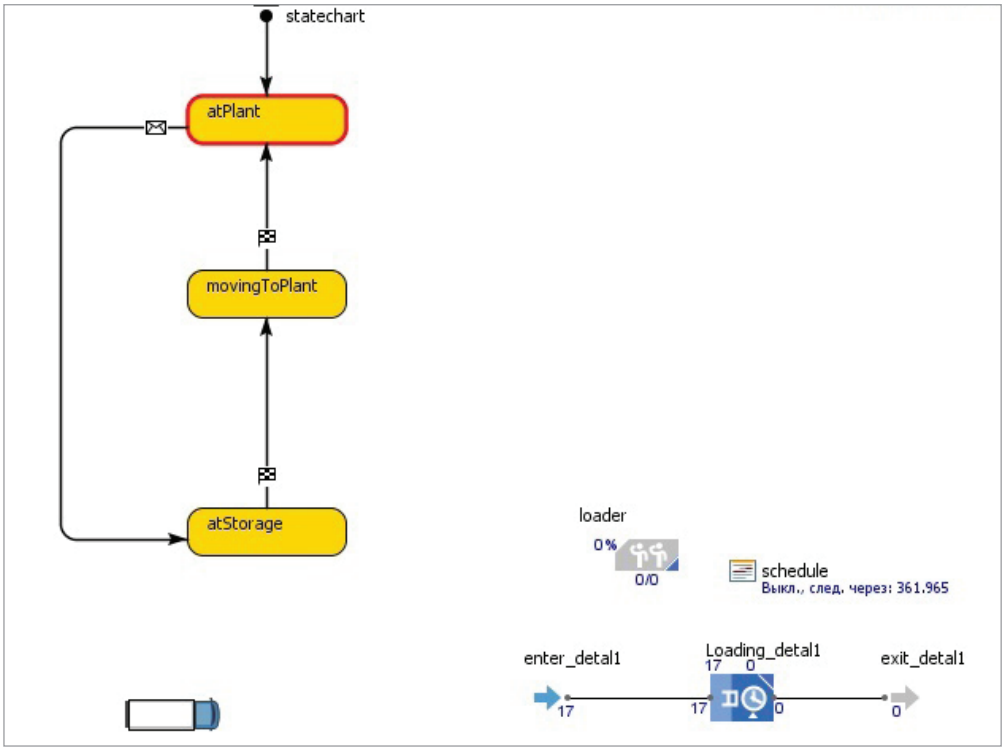

Рис. 6.54. Моделирование работы одного из грузовиков по доставке деталей со склада

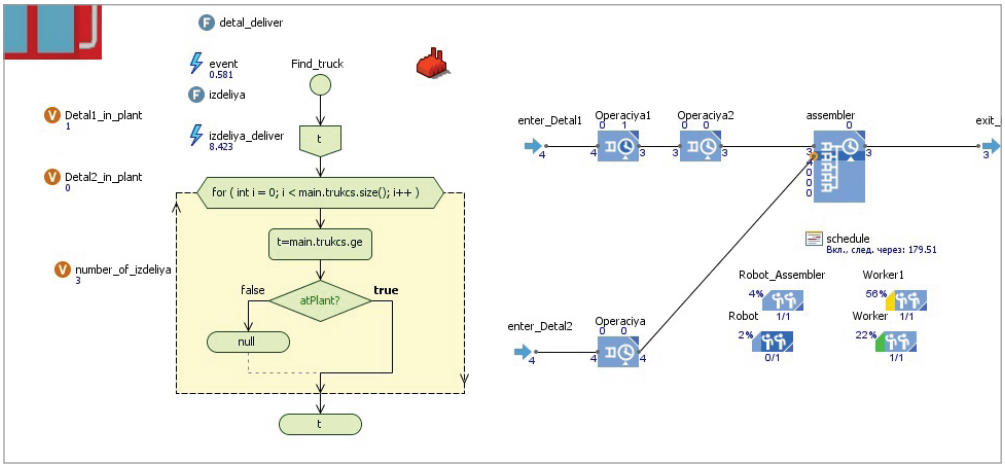

Рис. 6.55. Моделирование работы цеха по сборке изделий

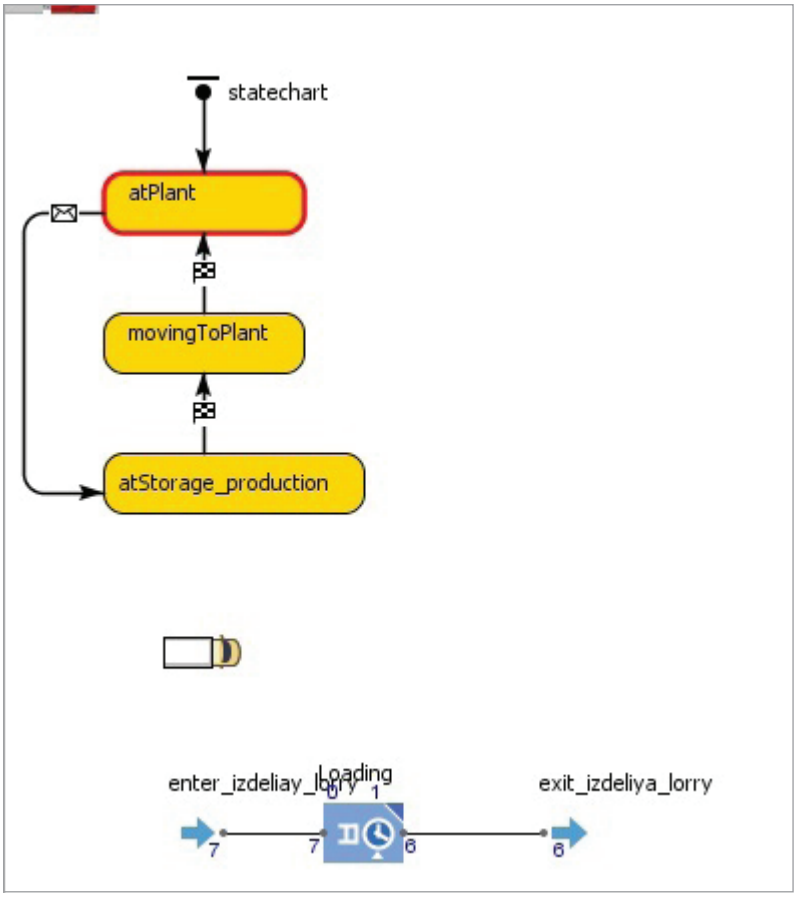

Рис. 6.56. Моделирование работы грузовика по доставке изделий на склад готовой продукции

Содержание

Для заметок

Учебное издание

Лимановская Оксана Викторовна,
Алферьева Татьяна Игоревна

МОДЕЛИРОВАНИЕ ПРОИЗВОДСТВЕННЫХ ПРОЦЕССОВ В ANYLOGIC 8.1

Редактор И. В. Коршунова
Верстка О. П. Игнатьевой

Подписано в печать 27.05.2019. Формат 70×100/16.
Бумага офсетная. Цифровая печать. Усл. печ. л. 11,0.
Уч.-изд. л. 6,0. Тираж 40 экз. Заказ 166.

Издательство Уральского университета
Редакционно-издательский отдел ИПЦ УрФУ
620049, Екатеринбург, ул. С. Ковалевской, 5
Тел.: +7 (343) 375-48-25, 375-46-85, 374-19-41
E-mail: rio@urfu.ru

Отпечатано в Издательско-полиграфическом центре УрФУ
620083, Екатеринбург, ул. Тургенева, 4
Тел.: +7 (343) 358-93-06, 350-58-20, 350-90-13
Факс: +7 (343) 358-93-06
http://print.urfu.ru


ЛИДЕР УНИВЕРСИТЕТСКОГО КНИГОИЗДАНИЯ
УНИВЕРСИТЕТСКАЯ КНИГА
2018


9 785799 626808